



Proceedings of the 12th International Conference on Educational Data Mining

Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds).

UQÀM | **Université du Québec
à Montréal**

**POLYTECHNIQUE
MONTRÉAL**

WORLD-CLASS
ENGINEERING



Proceedings of the 12th International Conference on Educational Data Mining
Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds).
July 2nd – 5th 2019 Montréal Canada
ISBN: 978-1-7336736-0-0

PREFACE

For this 12th iteration of the International Conference on Educational Data Mining (EDM 2019), the conference returned to its birthplace in Montreal, Quebec, Canada. EDM is organized under the auspices of the International Educational Data Mining Society in Montreal, Canada. The conference, held July 2nd through 5th, 2019, follows eleven previous editions (Buffalo 2018, Wuhan 2017, Raleigh 2016, Madrid 2015, London 2014, Memphis 2013, Chania 2012, Eindhoven 2011, Pittsburgh 2010, Cordoba, 2009 and Montreal 2008).

The theme of this year's conference is EDM in Open-Ended Domains. As EDM has matured it has increasingly been applied to open-ended and ill-defined tasks such as writing, design, and collaborative problem solving. And it has been used in new informal contexts where student actions are at best semi-structured. This iteration of the conference includes a range of work in these and other areas. This year's conference features three invited talks: Julita Vassileva, Professor at the Department of Computer Science, University of Saskatchewan, Canada; Steve Ritter, Co-Founder and Chief Scientist, Carnegie Learning Inc., Pittsburgh, USA; and Michael Mozer, Professor Department of Computer Science and Institute of Cognitive Science University of Colorado, USA.

This year's EDM conference has adopted a double-blind review process. Further, the program committee has been substantially extended, both to better reflect the community presenting works at EDM in the past years and to keep the review load for each member manageable. The number of papers submitted in the research track has increased compared to preceding years, resulting in 185 long and short paper submissions from 32 countries. Of these, 22 were accepted as full papers and 42 accepted as short papers. The combined acceptance rate of long and short papers is 34.6%. The acceptance rate for long papers is 21%. An additional 47 papers were accepted to the poster track. The poster and demo track itself accepted 14 contributions out of 34 submissions.

Together with the Journal of Educational Data Mining (JEDM), the EDM 2019 conference held a JEDM Track that provides researchers a venue to deliver more substantial mature work than is possible in a conference proceeding and to present their work to a live audience. The papers submitted to this track followed the JEDM peer review process. Seven papers were submitted and two papers are featured in the conference's program. Additionally this year, papers that were regularly published in the journal in 2018 were invited for presentation at the conference. Two authors accepted this invitation.

The main conference invited contributions to an Industry Track in addition to the main track. The EDM 2019 Industry Track received eleven submissions of which six were accepted. These figures are comparable, though slightly increasing, with last year's figures and suggest that connections between industry researchers and the academic research community become firmer.

The EDM conference continues its tradition of providing opportunities for young researchers to present their work and receive feedback from their peers and senior researchers. The doctoral consortium this year features eight such presentations.

Finally, stepping in the footsteps of last year's conference, this year's conference includes also an invited talk by the authors of the 2018 winner of the EDM Test of Time Award. This year's talk is delivered by Mykola Pechenizkiy.

In addition to the main program, there are three workshops: Learning Analytics: Building bridges between the Education and the Computing communities, Reinforcement Learning for Educational Data Mining, and Workshop on EDM & Games: Leveling Up Engaged Learning with Data-Rich Analytics, and three tutorials as well: Sharing and Reusing Data and Analytic Methods with LearnSphere, Causal Discovery with Tetrad in LearnSphere’s Tigris and Designing and Developing Open, Pedagogically-Based Predictive Models using the Moodle Analytics API.

We thank the sponsors of EDM 2019 for their generous support: Tourism Montréal, Prodigy, Ivado, SAS, Squirrel AI Learning, and ACT as well as our two host institutions: Université du Québec à Montréal (UQAM) and Polytechnique Montréal. We are also thankful to the senior program committee and regular program committee members and reviewers, without whose expert input this conference would not be possible. Finally, we thank the entire organizing team and all authors who submitted their work to EDM 2019. And we thank EasyChair for their infrastructural support.

<i>Collin F. Lynch</i>	North Carolina State University,	Program Chair
<i>Agathe Merceron</i>	Beuth University of Applied Sciences Berlin,	Program Chair
<i>Michel Desmarais</i>	Polytechnique Montréal,	General Chair
<i>Roger Nkambou</i>	Université du Québec à Montréal,	General Chair

July 2nd, 2019
Montreal, QC, Canada.

ORGANIZING COMMITTEE

General chairs: Michel Desmarais (Polytechnique Montréal) and Roger Nkambou (Université du Québec à Montréal)

Program chairs: Collin F. Lynch (North Carolina State University) and Agathe Merceron (Beuth University of Applied Sciences Berlin)

Workshop & Tutorial Chairs: Luc Paquette (The University of Illinois at Urbana-Champaign) and Cristobol Romero (University of Cordoba)

Industry Track Chairs: KP Thai (Yixue Education) and Mary Jean Blink (TutorGen)

Doctoral Consortium Chairs: Andrew Lan (Princeton University) and Mykola Pechenizkiy (Eindhoven University of Technology, TU/e)

JEDM Track Chairs: Michael Yudelson (ACT, Inc.) and Andrew Olney (University of Memphis)

Poster & Demo Track Chairs: Bruno Emond (National Research Council of Canada) and Jill-Jenn Vie (RIKEN, Tokyo)

Sponsorship Chairs: Steven Fancsali (Carnegie Learning, Inc., Pittsburgh) and Roger Nkambou (Université du Québec à Montréal)

Volunteer Chairs: Sophie Calliès and Ange Tato (Université du Québec à Montréal)

Publicity/Social Media Chair: Michael Eagle (Carnegie-Mellon University)

Web chair: Paul Salvador Inventado (California State University, Fullerton).

IEDMS Officers:

Mykola Pechenizkiy,	President	Eindhoven University of Technology, Netherlands
Kenneth Koedinger,	President-elect	Carnegie Mellon University, USA
Mingyu Feng,	Treasurer	WestEd, USA

IEDMS Board of Directors:

Rakesh Agrawal	Data Insights Laboratories, USA
Ryan Baker	University of Pennsylvania, USA
Tiffany Barnes	North Carolina State University, USA
Michel Desmarais	Ecole Polytechnique de Montreal, Canada
Sidney D'Mello	University of Colorado Boulder, USA
Luc Paquette	University of Illinois Urbana-Champaign, USA
John Stamper	Carnegie Mellon University, USA
Kalina Yacef	University of Sydney, Australia

Program Committee

Lalitha Agnihotri	McGraw Hill Education
Rakesh Agrawal	Data Insights Laboratories
Esma Aimeur	University of Montreal
Bitu Akram	North Carolina State University
Giora Alexandron	Weizmann Institute of Science
Boyer Anne	Kiwi Team - LORIA
Mirjam Augstein	Upper Austria University of Applied Sciences, Communication and Knowledge Media
Roger Azevedo	University of Central Florida
Costin Badica	University of Craiova, Computer and Information Technology Department, Romania
Ryan Baker	University of Pennsylvania
Richard Baraniuk	Rice University
Tiffany Barnes	North Carolina State University
Nabil Belacel	National Research Council
Suma Bhat	University of Illinois at Urbana-Champaign
Gautam Biswas	Vanderbilt University
Mary Jean Blink	TutorGen, Inc.
Nigel Bosch	University of Illinois Urbana-Champaign
Jesus G. Boticario	UNED
François Bouchet	Sorbonne Université - LIP6
Alex Bowers	Columbia University
Kristy Elizabeth Boyer	University of Florida
Javier Bravo Agapito	Madrid Open University (UDIMA)
Keith Brawner	United States Army Research Laboratory
Christopher Brinton	Princeton University
Armelle Brun	LORIA - Université Nancy 2
Peter Brusilovsky	University of Pittsburgh
Zhiqiang Cai	The University of Memphis
Renza Campagni	Università degli Studi di Firenze
Alberto Cano	Virginia Commonwealth University
Ted Carmichael	TutorGen, Inc.
Zhongzhou Chen	University of Central Florida Physics Department
Min Chi	North Carolina State University
Mihaela Cocea	School of Computing, University of Portsmouth
Scott Crossley	Georgia State University
Cynthia D'Angelo	University of Illinois at Urbana-Champaign
Sidney D'Mello	University of Colorado Boulder
Michel Desmarais	Ecole Polytechnique de Montreal
Prasenjit Dey	IBM
Shayan Doroudi	Carnegie Mellon University
Kerrie Douglas	Purdue University
Guillaume Durand	National Research Council Canada
Michael Eagle	Carnegie Mellon University
Asmaa Elbadrawy	Arizona State University

Bruno Emond	National Research Council Canada
Collin F. Lynch	North Carolina State University
Stephen Fancsali	Carnegie Learning, Inc.
Scott Farrar	Khan Academy
Mingyu Feng	SRI International
Jeremiah Folsom-Kovarik	Soar Technology, Inc.
Carol Forsyth	Educational Testing Service
Davide Fossati	Emory University
Kobi Gal	Ben Gurion University
April Galyardt	Software Engineering Institute
Dragan Gasevic	Monash University
Niki Gitinabard	North Carolina State University
Lukasz Golab	University of Waterloo
Ilya Goldin	2U, Inc.
José González-Brenes	Chegg
Cyril Goutte	National Research Council Canada
Joseph Grafsgaard	ACM
Nathalie Guin	LIRIS - Université de Lyon
Jiangang Hao	Educational Testing Service
Erik Harpstead	Carnegie Mellon University
Yugo Hayashi	Ritsumeikan University
Neil Heffernan	Worcester Polytechnic Institute
Arnon HersHKovitz	Tel Aviv University
Sharon Hsiao	Arizona State University
Xiangen Hu	The University of Memphis
Yuchi Huang	ACT Inc
Vladimir Ivančević	University of Novi Sad, Faculty of Technical Sciences
Jelena Jovanovic	University of Belgrade
Fengfeng Ke	Florida State University
Farzaneh Khoshnevisan	North Carolina State University
Hassan Khosravi	The University of Queensland
Kenneth Koedinger	Carnegie Mellon University
Irena Koprinska	The University of Sydney
Sotiris Kotsiantis	University of Patras
Christopher Krauss	Fraunhofer
Sébastien Lallé	The University of British Columbia, Department of Computer Science
Andrew Lan	University of Massachusetts Amherst
Charles Lang	Columbia University
James Lester	North Carolina State University
Chun Wai Liew	Lafayette College
Chen Lin	North Carolina State University
Christina Lioma	University of Copenhagen
Ran Liu	Carnegie Mellon University
Vanda Luengo	Sorbonne Université - LIP6
Ivan Luković	University of Novi Sad, Faculty of Technical Sciences
Ling Luo	University of Technology, Sydney

Maria Luque	University of Cordoba
Christopher Maclellan	Soar Technology, Inc.
Patricia Mahabir	MREI Inc.
Noboru Matsuda	North Carolina State University
Manolis Mavrikis	London Knowledge Lab
Victor Menendez	Universidad Autónoma de Yucatán
Agathe Merceron	Beuth Hochschule für Technik Berlin
Donatella Merlini	Università di Firenze
Cristian Mihaescu	University of Craiova
Wookhee Min	North Carolina State University
Piotr Mitros	ETS / MIT
Bradford Mott	North Carolina State University
Benjamin Motz	Indiana University Bloomington
Michael Mozer	University of Colorado Boulder
Chas Murray	Carnegie Learning, Inc.
Roger Nkambou	Université du Québec À Montréal (UQAM)
Andrew Olney	University of Memphis
Shai Olsher	University of Haifa
Elizabeth Owen	Age of Learning, Inc.
Andreas Paepcke	Stanford University
Luc Paquette	University of Illinois at Urbana-Champaign
Abelardo Pardo	University of South Australia
Zach Pardos	University of California, Berkeley
Philip I. Pavlik Jr.	University of Memphis
Mykola Pechenizkiy	Eindhoven University of Technology
Radek Pelánek	Masaryk University Brno
Niels Pinkwart	Humboldt-Universität zu Berlin
Agoritsa Polyzou	University of Minnesota
Paul Stefan Popescu	University of Craiova, Faculty of Automation, Computers and Electronics, Craiova, Romania
Thomas Price	North Carolina State University
David Pritchard	Massachusetts Institute of Technology
Shi Pu	Purdue University
Huzefa Rangwala	George Mason University
Martina Rau	University of Wisconsin - Madison, Department of Educational Psychology
Steve Ritter	Carnegie Learning, Inc.
Ma. Mercedes T. Rodrigo	Department of Information Systems and Computer Science, Ateneo de Manila University
José Raúl Romero	University of Cordoba
Cristobal Romero Morales	University of Cordoba
Carolyn Rose	Carnegie Mellon University
Vasile Rus	The University of Memphis
Shaghayegh Sahebi	University at Albany - SUNY
Adam Sales	The University of Texas at Austin
Maria Ofelia San Pedro	ACT, Inc.
Sweet San Pedro	ACT, Inc.

Olga C. Santos	aDeNu Research Group (UNED)
David Shaffer	University of Wisconsin-Madison
Shitian Shen	North Carolina State University
Erica Snow	Imbellus, Inc.
John Stamper	Carnegie Mellon University
Jun-Ming Su	Department of Information and Learning Technology, National University of Tainan
Ling Tan	Australian Council for Educational Research
Kp Thai	Squirrel AI Learning
Stefan Trausan-Matu	University Politehnica of Bucharest
Jennifer Tsan	North Carolina State University
Sebastián Ventura	University of Cordoba. Dept. of Computer Science and Numerical Analysis
Jill-Jênn Vie	RIKEN AIP
Lucian Vintan	"Lucian Blaga" University of Sibiu
Alina von Davier	ACTNext
Elle Yuan Wang	Arizona State University
Stephan Weibelzahl	Private University of Applied Sciences Göttingen
Jacob Whitehill	Worcester Polytechnic Institute
Eric Wiebe	North Carolina State University
Joseph Wiggins	University of Florida
Verena Wolf	Saarland University
Beverly Park Woolf	University of Massachusetts
Linting Xue	North Carolina State University
Kalina Yacef	The University of Sydney
Amelia Zafra Gómez	Department of Computer Sciences and Numerical Analysis
Alfredo Zapata González	Universidad Autonoma de Yucatan
Diego Zapata-Rivera	Educational Testing Service
Chengxiang Zhai	University of Illinois at Urbana-Champaign
Guojing Zhou	North Carolina State University
Craig Zilles	University of Illinois at Urbana-Champaign
Amal Zouaq	Polytechnique

SPONSORS



Best Paper Selection

The conference chairs selected the three best papers based upon the reviews and meta-reviews for each paper. The papers were then sent to an outside awards committee. Each committee member read and ranked the three papers. The best paper award was attributed to the highest ranked paper. The best student paper was awarded to the next highest ranked paper in the set.

Best Paper Award Committee

- Rakesh Agrawal
- Ryan Baker
- Andrew Olney
- Luc Paquette
- Kalina Yacef

Best Paper Nominees

Benoît Choffin, Fabrice Popineau, Yolaine Bourda and Jill-Jänn Vie. *DAS3H: a new student learning and forgetting model for optimally scheduling distributed practice of skills* page 39.

Markel Sanz Ausin, Hamoon Azizsoltani, Tiffany Barnes and Min Chi. *Leveraging Deep Reinforcement Learning for Pedagogical Policy Induction in an Intelligent Tutoring System* page 178.

Ye Mao, Rui Zhi, Farzaneh Khoshnevisan, Thomas Price, Tiffany Barnes and Min Chi. *One minute is enough: Early Prediction of Student Success and Event-level Difficulty during Novice Programming Tasks* page 129.

TABLE OF CONTENTS

Keynotes

On the Ethics of Data-Powered Instruction, Recommendation, Persuasion, and Behaviour Change.....	1
--	---

Julita Vassileva

Reconsidering two sigma: Educational data mining for the complete instructional system.....	3
---	---

Steve Ritter

Educating a synthetic student	4
-------------------------------------	---

Michael Mozer

JEDM Presentations

Predictiveness of Prior Failures is Improved by Incorporating Trial Duration	5
--	---

Luke G. Eglington and Philip I. Pavlik Jr.

Will this Course Increase or Decrease Your GPA? Towards Grade-aware Course Recommendation	6
---	---

Sara Morsy and George Karypis

The Continuous Hint Factory - Providing Hints in Vast and Sparsely Populated Edit Distance Spaces	7
---	---

Benjamin Paaßen, Barbara Hammer, Thomas Price, Tiffany Barnes, Sebastian Gross and Niels Pinkwart

Using Sequence Mining to Analyze Metacognitive Monitoring and Scientific Inquiry based on Levels of Efficiency and Emotions during Game-Based Learning.....	8
---	---

Michelle Taub and Roger Azevedo

Full Papers

Mining University Registrar Records to Predict First-Year Undergraduate Attrition	9
<i>Lovenoor Aulck, Dev Nambi, Nishant Velagapudi, Joshua Blumenstock and Jevin West</i>	
Predictors of Student Satisfaction: A Large-scale Study of Human-Human Online Tutorial Dialogues	19
<i>Guanliang Chen, David Lang, Rafael Ferreira and Dragan Gasevic</i>	
DAS3H: Modeling Student Learning and Forgetting for Optimally Scheduling Distributed Practice of Skills	29
<i>Benoît Choffin, Fabrice Popineau, Yolaine Bourda and Jill-Jênn Vie</i>	
Predicting Early and Often: Predictive Student Modeling for Block-Based Programming Environments	39
<i>Andrew Emerson, Andy Smith, Cody Smith, Fernando Rodríguez, Wookhee Min, Eric Wiebe, Bradford Mott, Kristy Boyer and James Lester</i>	
Modeling and Experimental Design for MOOC Dropout Prediction: A Replication Perspective	49
<i>Josh Gardner, Yuming Yang, Ryan Baker and Christopher Brooks</i>	
What will you do next? A Sequence Analysis of the Student Transitions between Online Platforms	59
<i>Niki Gitinabard, Sarah Heckman, Tiffany Barnes and Collin Lynch</i>	
Academic Performance Estimation with Attention-based Graph Convolutional Networks	69
<i>Qian Hu and Huzefa Rangwala</i>	
Evaluating Fairness and Generalizability in Models Predicting On-Time Graduation from College Applications	79
<i>Stephen Hutt, Margo Gardner, Angela L. Duckworth and Sidney D’Mello</i>	
Measuring students’ thermal comfort and its impact on learning	89
<i>Han Jiang, Matthew Iandoli, Steven Van Dessel, Shichao Liu and Jacob Whitehill</i>	
The Influence of School Demographics on the Relationship Between Students’ Help-Seeking Behavior and Performance and Motivational Measures	99
<i>Shamya Karumbaiah, Jaclyn Ocumpaugh and Ryan S Baker</i>	

Exploring Neural Network Models for the Classification of Students in Highly Interactive Environments.....	109
<i>Tanja Käser and Daniel L. Schwartz</i>	
One minute is enough: Early Prediction of Student Success and Event-level Difficulty during Novice Programming Tasks.....	119
<i>Ye Mao, Rui Zhi, Farzaneh Khoshnevisan, Thomas Price, Tiffany Barnes and Min Chi</i>	
Kappa Learning: A New Item-Similarity Method for Clustering Educational Items from Response Data	129
<i>Tanya Nazaretsky, Sara HersHKovitz and Giora Alexandron</i>	
Using Knowledge Component Modeling to Increase Domain Understanding in a Digital Learning Game.....	139
<i>Huy Nguyen, Yeyu Wang, John Stamper and Bruce McLaren</i>	
Predicting the Quality of Collaborative Problem Solving Through Linguistic Analysis of Discourse.....	149
<i>Joseph Reilly and Bertrand Schneider</i>	
Grade Prediction Based on Cumulative Knowledge and Co-taken Courses	158
<i>Zhiyun Ren, Xia Ning, Andrew Lan and Huzefa Rangwala</i>	
Leveraging Deep Reinforcement Learning for Pedagogical Policy Induction in an Intelligent Tutoring System.....	168
<i>Markel Sanz Ausin, Hamoon Azizsoltani, Tiffany Barnes and Min Chi</i>	
Optimizing Assignment of Students to Courses based on Learning Activity Analytics.....	178
<i>Atsushi Shimada, Kousuke Mouri, Yuta Taniguchi, Hiroaki Ogata, Rin-Ichiro Taniguchi and Shinichi Konomi</i>	
Towards the prediction of semantic complexity based on concept graphs.....	188
<i>Rémi Venant and Mathieu d'Aquin</i>	
Affective State Prediction in a Mobile Setting using Wearable Biometric Sensors and Stylus ...	198
<i>Rafael Wampfler, Severin Klingler, Barbara Solenthaler, Victor Schinazi and Markus Gross</i>	

Active Learning for Student Affect Detection	208
<i>Tsung-Yen Yang, Christoph Studer, Ryan S. Baker, Neil T. Heffernan and Andrew Lan</i>	
Toward Data-Driven Example Feedback for Novice Programming	218
<i>Rui Zhi, Samiha Marwan, Yihuan Dong, Nicholas Lytle, Thomas Price and Tiffany Barnes</i>	
Short Papers	
A Multivariate ELO-based Learner Model for Adaptive Educational Systems	228
<i>Solmaz Abdi, Hassan Khosravi, Shazia Sadiq and Dragan Gasevic</i>	
How Should Online Teachers of English as a Foreign Language (EFL) Write Feedback to Students?	234
<i>Cecilia Aguerrebere, Monica Bulger, Cristóbal Cobo, Sofía García, Gabriela Kaplan and Jacob Whitehill</i>	
Concept-Aware Deep Knowledge Tracing and Exercise Recommendation in an Online Learning System	240
<i>Fangzhe Ai, Yishuai Chen, Yuchun Guo, Yongxiang Zhao, Zhenzhu Wang, Guowei Fu and Guangyan Wang</i>	
Clustering Students Based on Their Prior Knowledge	246
<i>Nisrine Ait Khayi and Vasile Rus</i>	
Grades are not Normal: Improving Exam Score Models Using the Logit-Normal Distribution ..	252
<i>Noah Arthurs, Ben Stenhaus, Sergey Karayev and Chris Piech</i>	
Assessing Student Response in Tutorial Dialogue Context using Probabilistic Soft Logic	258
<i>Rajendra Banjade and Vasile Rus</i>	
Application of Hidden Markov Models to quantify the impact of enrollment patterns on student performance	264
<i>Shahab Boumi and Adan Vela</i>	
Design and evaluation of a semantic indicator for automatically supporting programming learning	270
<i>Julien Broisin and Clément Hérouard</i>	

Exploring the Link Between Motivations and Gaming.....	276
<i>Steven Dang and Kenneth Koedinger</i>	
Why Deep Knowledge Tracing has less Depth than Anticipated.....	282
<i>Xinyi Ding and Eric Larson</i>	
Rank-Based Tensor Factorization for Predicting Student Performance.....	288
<i>Thanh-Nam Doan and Shaghayegh Sahebi</i>	
Using Latent Variable Models to Observe Academic Pathways.....	294
<i>Nate Gruver, Ali Malik, Brahm Capoor, Chris Piech, Mitchell Stevens and Andreas Paepcke</i>	
A Human-Machine Hybrid Peer Grading Framework for SPOCs.....	300
<i>Yong Han, Wenjun Wu, Suozhao Ji, Lijun Zhang and Hui Zhang</i>	
Modelling End-of-Session Actions in Educational Systems.....	306
<i>Christian Hansen, Casper Hansen, Stephen Alstrup and Christina Lioma</i>	
Categorizing students' questions using an ensemble hybrid approach.....	312
<i>Fatima Harrak, François Bouchet and Vanda Luengo</i>	
Identifying Collaborative Learning States Using Unsupervised Machine Learning on Eye-Tracking, Physiological and Motion Sensor Data.....	318
<i>Karina Huang, Tonya Bryant and Bertrand Schneider</i>	
Generalizability of Sensor-Free Affect Detection Models in a Longitudinal Dataset of Tens of Thousands of Students.....	324
<i>Emily Jensen, Stephen Hutt and Sidney K. D'Mello</i>	
Time-series Insights into the Process of Passing or Failing Online University Courses using Neural-Induced Interpretable Student States.....	330
<i>Byungsoo Jeon, Eyal Shafran, Luke Breitfeller, Jason Levin and Carolyn P. Rosé</i>	
Collaborative problem-solving process in a science serious game: Exploring Group Action Similarity Trajectory.....	336
<i>Jina Kang, Dongwook An, Lili Yan and Min Liu</i>	

Generalizing Expert Misconception Diagnoses Through Common Wrong Answer Embedding ..	342
<i>John Kolb, Scott Farrar and Zachary A. Pardos</i>	
Active Learning of Strict Partial Orders: A Case Study on Concept Prerequisite Relations	348
<i>Chen Liang, Jianbo Ye, Han Zhao, Bart Pursel and C. Lee Giles</i>	
Characterising Students' Writing Processes Using Temporal Keystroke Analysis	354
<i>Donia Malekian, James Bailey, Gregor Kennedy, Paula de Barba and Sadia Nawaz</i>	
Skills Embeddings: a Neural Approach to Multicomponent Representations of Students and Tasks	360
<i>Russell Moore, Andrew Caines, Mark Elliott, Ahmed Zaidi, Andrew Rice and Paula Buttery</i>	
Sparse Neural Attentive Knowledge-based Model for Grade Prediction	366
<i>Sara Morsy and George Karypis</i>	
Modeling person-specific development of math skills in continuous time: New evidence for mutualism	372
<i>Lu Ou, Abe Hofman, Vanessa Simmering, Timo Bechger, Gunter Maris and Han van der Maas</i>	
Detecting Wheel Spinning and Productive Persistence in Educational Games	378
<i>V. Elizabeth Owen, Marie-Helene Roy, K. P. Thai, Vesper Burnett, Daniel Jacobs, Eric Keylor and Ryan S. Baker</i>	
A Self Attentive model for Knowledge Tracing	384
<i>Shalini Pandey and George Karypis</i>	
Success prediction in MOOCs - A case study	390
<i>Antoine Pigeau, Olivier Aubert and Yannick Prié</i>	
Scholars Walk: A Markov Chain Framework for Course Recommendation	396
<i>Agoritsa Polyzou, Athanasios N. Nikolakopoulos and George Karypis</i>	
A Comparison of Automated Scale Short Form Selection Strategies	402
<i>Anthony Raborn, Walter Leite and Katerina Marcoulides</i>	

Detecting Creativity in an Open Ended Geometry Environment	408
<i>Roi Shillo, Nicholas Hoernle and Kobi Gal</i>	
Tutorbot Corpus: Evidence of Human-Agent Verbal Alignment in Second Language Learner Dialogues	414
<i>Arabella Sinclair, Kate McCurdy, Adam Lopez, Christopher G. Lucas and Dragan Gasevic</i>	
Utterance-level Modeling of Indicators of Engaging Classroom Discourse	420
<i>Cathlyn Stone, Patrick Donnelly, Meghan Dale, Sarah Capello, Sean Kelly, Amanda Godley and Sidney K. D'Mello</i>	
Probabilistic Modeling of Peer Correction and Peer Assessment	426
<i>Takeru Sunahase, Yukino Baba and Hisashi Kashima</i>	
Implicit and Explicit Emotions in MOOCs	432
<i>Munira Syed, Malolan Chetlur, Shazia Afzal, G. Alex Ambrose and Nitesh V. Chawla</i>	
Planning Gamification Strategies based on User Characteristics and DM: A Gender-based Case Study	438
<i>Armando Maciel Toda, Wilk Oliveira, Lei Shi, Ig Ibert Bittencourt, Seiji Isotani and Alexandra Cristea</i>	
Grading emails and generating feedback	444
<i>Abhishek Unnam, Rohit Takhar and Varun Aggarwal</i>	
Improving Peer Assessment Accuracy by Incorporating Relative Peer Grades	450
<i>Tianqi Wang, Qi Li, Jing Gao, Xia Jing and Jie Tang</i>	
Toward Near Zero-Parameter Prediction Using a Computational Model of Student Learning ...	456
<i>Daniel Weitekamp, Erik Harpstead, Napol Rachatasumrit, Christopher Maclellan and Kenneth R. Koedinger</i>	
Do Learners Know What's Good for Them? Crowdsourcing Subjective Ratings of OERs to Predict Learning Gains	462
<i>Jacob Whitehill, Cecilia Aguerrebere and Benjamin Hylak</i>	

Early detection of wheel spinning: Comparison across tutors, models, features, and operationalizations	468
<i>Chuankai Zhang, Yanzun Huang, Jingyu Wang, Dongyang Lu, Weiqi Fang, John Stamper, Stephen Fancsali, Kenneth Holstein and Vincent Alevan</i>	
Detecting suggestions in peer assessments	474
<i>Gabriel Zingle, Balaji Radhakrishnan, Yunkai Xiao, Edward Gehringer, Zhongcan Xiao, Ferry Pramudianto, Gauraang Khurana and Ayush Arnav</i>	
Posters	
Towards a General Purpose Anomaly Detection Method to Identify Cheaters in Massive Open Online Courses	480
<i>Giora Alexandron, José A. Ruipérez-Valiente and David Pritchard</i>	
A Novel Use of Educational Data Mining to Inform Effective Management of Academic Programs	484
<i>Anwar Ali Yahya, Fekri Mohammed and Addin Osman</i>	
Assessing the Fairness of Graduation Predictions	488
<i>Henry Anderson, Afshan Boodhwani and Ryan Baker</i>	
Hello? Who is posting, who is answering, and who is succeeding in Massive Open Online Courses	492
<i>Juan Miguel Andres-Bray, Jaclyn Ocumpaugh and Ryan S. Baker</i>	
Augmenting Transcripts with Natural Language Processing and Multimodal Data	496
<i>Tyler Angert and Bertrand Schneider</i>	
Predicting Student Dropout in Higher Education Based on Previous Exam Results	500
<i>Alexander Askinadze and Stefan Conrad</i>	
The Guided Team-Partitioning Problem: Definition, Complexity, and Algorithm	504
<i>Sanaz Bahargam, Theodoros Lappas and Evimaria Terzi</i>	
Machine-Learned or Expert-Engineered Features? Exploring Feature Engineering Methods in Detectors of Disengaged Behavior and Affect	508
<i>Anthony F. Botelho, Ryan S. Baker and Neil T. Heffernan</i>	

Shedding Light on the Automated Essay Scoring Process.....	512
<i>David Boulanger and Vivekanandan Kumar</i>	
Incorporating Prior Practice Difficulty into Performance Factors Analysis to Model Mandarin Tone Learning.....	516
<i>Meng Cao, Philip Pavlik and Gavin Bidelman</i>	
Parent as a Companion for Solving Challenging Math Problems: Insights from Multi-modal Observational Data.....	520
<i>Lujie Chen, Eva Gjekmarkaj and Artur Dubrawski</i>	
Gender Differences in Work-Integrated Learning Assessments.....	524
<i>Shivangi Chopra, Abeer Khan, Melicaalsadat Mirsafian and Lukasz Golab</i>	
Individual Differences in Student Learning Aid Usage.....	528
<i>Andrea Davis, Yun Jin Rho and Daniel Furr</i>	
N-gram Graphs for Topic Extraction in Educational Forums.....	532
<i>Glenn Davis, Cindy Wang and Christina Yuan</i>	
A Data-Driven Approach for Automated Assessment of Scientific Explanations in Science Inquiry.....	536
<i>Rachel Dickler, Haiying Li and Janice Gobert</i>	
Design and Deployment of a Better University Course Search: Inferring Latent Keywords from Enrollments.....	540
<i>Matthew Dong, Run Yu and Zachary Pardos</i>	
Visualizing Learning Performance Data and Model Predictions as Objects in a 3D Space.....	544
<i>Bruno Emond and Julio J. Valdés</i>	
A generalizable performance evaluation model of driving games via risk-weighted trajectories..	548
<i>Rory Flemming, Emmanuel Schmück, Dominic Mussack, Pedro Cardoso-Leite and Paul Schrater</i>	
Visualization and clustering of learner pathways in an interactive online learning environment.	552
<i>Daniel Furr</i>	

Filtering non-relevant short answers in peer learning applications.....	556
<i>Vincent Gagnon, Audrey Labrie, Michel Desmarais and Sameer Bhatnagar</i>	
Adding duration-based quality labels to learning events for improved description of students' online learning behavior	560
<i>Matthew Guthrie and Zhongzhou Chen</i>	
Automatic identification of questions in MOOC forums and association with self-regulated learning	564
<i>Fatima Harrak, François Bouchet, Vanda Luengo and Rémi Bachelet</i>	
Students' Use of Support Functions in DBAs: Analysis of NAEP Grade 8 Mathematics Process Data	568
<i>Juanita Hicks, Ruhan Circi and Mengyi Li</i>	
Investigating Writing Style Development in High School.....	572
<i>Niklas Hjuler, Stephan Lorenzen and Stephen Alstrup</i>	
Learning Feature Analysis for Quality Improvement of Web-Based Teaching Materials Using Mouse Cursor Tracking.....	576
<i>Mizuho Ikeda</i>	
It's a Match! Reciprocal Recommender System for Graduating Students and Jobs.....	580
<i>Anik Jacobsen and Gerasimos Spanakis</i>	
Supporting Minority Student Success by using Machine Learning to Identify At-Risk Students	584
<i>J.D Jayaraman, Sue Gerber and Julian Garcia</i>	
Binary Q-matrix Learning with dAFM.....	588
<i>Nan Jiang and Zachary Pardos</i>	
A Comparative Analysis of Emotional Words for Learning Effectiveness in Online Education ..	591
<i>Jaechoon Jo, Yeongwook Yang, Gyeongmin Kim and Heuiseok Lim</i>	
Identify Critical Pedagogical Decisions through Adversarial Deep Reinforcement Learning	595
<i>Song Ju, Guojing Zhou, Hamoon Azizzoltani, Tiffany Barnes and Min Chi</i>	

Smart Learning Object Recommendations based on Time-Dependent Learning Need Models ..	599
<i>Christopher Krauss, Agathe Merceron and Stefan Arbanowski</i>	
Teacher vs. algorithm double-blind experiment of content sequencing in mathematics	603
<i>Ben Levy, Arnon HersHKovitz, Odelya Tzayada, Orit Ezra, Avi Segal, Kobi Gal, Anat Cohen and Michal Tabach</i>	
Studying Factors Influencing the Prediction of Student STEM and Non-STEM Career Choice .	607
<i>Varun Mandalapu and Jiaqi Gong</i>	
A Methodology for Student Video Interaction Patterns Analysis and Classification	611
<i>Boniface Mbouzao, Michel Desmarais and Ian Shrier</i>	
Discovering item similarity through deep learning: combining item features and user behavior.	615
<i>Dominic Mussack, Rory Flemming, Paul Schrater and Pedro Cardoso-Leite</i>	
Machine Learning Based Decision Support System for Categorizing MOOC Discussion Forum Posts	619
<i>Gaurav Nanda and Kerrie Douglas</i>	
Hybrid Deep Neural Networks to Predict Socio-Moral Reasoning skills	623
<i>Ange Adrienne Nyamen Tato, Roger Nkambou and Aude Dufresne</i>	
Identifying bias and underlying knowledge structures in Brazilian National Exam of Students Performance	627
<i>Mariana Oliveira and Carlos E. Mello</i>	
Validating the Myth of Average through Evidences	631
<i>Praseeda, Srinath Srinivasa and Prasad Ram</i>	
ATC Framework: A fully Automatic Cognitive Tracing Model for Student and Educational Contents	635
<i>YanJun Pu, Wenjun Wu and Tianrui Jiang</i>	
Toward Instrumenting Makerspaces: Using Motion Sensors to Capture Students' Affective States in Open- Ended Learning Environments.....	639
<i>Lucia Ramirez, William Yao, Edwin Chng, Iulian Radu and Bertrand Schneider</i>	

Exploring Stealth Assessment via Deep Learning in an Open-Ended Virtual Environment	643
<i>Joseph Reilly and Chris Dede</i>	
Balancing Student Success and Inferring Personalized Effects in Dynamic Experiments	647
<i>Hammad Shaikh, Arghavan Modiri, Joseph Jay Williams and Anna Rafferty</i>	
Investigating effects of considering mobile and desktop learning data on predictive power of learning management system (LMS) features on student success	651
<i>Varshita Sher, Marek Hatala and Dragan Gasevic</i>	
Investigating Error Resolution Processes in C Programming Exercise Courses	655
<i>Yuta Taniguchi, Atsushi Shimada and Shin'Ichi Konomi</i>	
Staying in the Zone: Sequencing Content in Classrooms Based on the Zone of Proximal Development	659
<i>Oded Vainas, Yossi Ben-David, Ran Gilad-Bachrach, Meitar Ronen, Ori Bar-Ilan, Roi Shillo, Galit Lukin and Daniel Sitton</i>	
Hǎo Fā Yīn: Developing Automated Audio Assessment Tools for a Chinese Language Course . .	663
<i>Ashvini Varatharaj, Anthony Botelho, Xiwen Lu and Neil T. Heffernan</i>	
A Meta-Learning Approach to Automatic Short Answer Grading	667
<i>Zichao Wang, Andrew Lan, Andrew Waters, Phillip Grimaldi and Richard Baraniuk</i>	
Deep Hierarchical Knowledge Tracing	671
<i>Tianqi Wang, Fenglong Ma and Jing Gao</i>	
Beyond Autoscoring: Extracting Conceptual Connections from Essays for Classroom Instruction	675
<i>Korah Wiley, Allison Bradford, Zach Pardos and Marcia Linn</i>	
What You Say is Relevant to How You Make Friends: Measuring the Effect of Content on Social Connection	679
<i>Yiqiao Xu, Niki Gitinabard, Collin Lynch and Tiffany Barnes</i>	

Deep-IRT: Make Deep Learning Based Knowledge Tracing Explainable Using Item Response Theory	683
---	-----

Chun-Kit Yeung

Accurate Modelling of Language Learning Tasks and Students Using Representations of Grammatical Proficiency.....	687
--	-----

Ahmed Zaidi, Andrew Caines, Christopher Davis, Russell Moore, Paula Buttery and Andrew Rice

Student Knowledge Diagnosis on Response Data via the Model of Sparse Factor Learning	691
--	-----

Yupei Zhang, Huan Dai, Yue Yun and Xuequn Shang

Doctoral Consortium

Collaboration Analysis Using Object Detection	695
---	-----

Zhang Guo and Roghayeh Barmaki

Design of an Elective Course Recommendation System for University Environment	699
---	-----

Boxuan Ma

Towards Modeling Students' Problem-solving Skills in Non-routine Mathematics Problems	702
--	-----

Huy Nguyen, John Stamper and Bruce McLaren

Anatomy of mobile learners: Using learning analytics to unveil learning in presence of mobile devices	706
---	-----

Varshita Sher

Modeling Student Performance and Disengagement Using Decomposition of Response Time Data	710
--	-----

Deniz Sonmez Unal

Techniques for Automatically Evaluating Machine-Authored Homework Questions	714
---	-----

Zichao Wang

Beyond Autoscoring: Extracting Conceptual Connections from Essays for Classroom Instruction	718
---	-----

Korah Wiley, Allison Bradford, Zach Pardos and Marcia Linn

Predicting student academic outcomes in UK secondary phase education: an architecture for machine learning and user interaction.....	722
--	-----

Matthew Woodruff

Industry Track

Machine-Learned School Dropout Early Warning at Scale.....	726
--	-----

S. Thomas Christie, Daniel Jarratt, Lukas Olson and Taavi Taijala

A Better Cold-Start for Early Prediction of Student At-Risk Status in New School Districts...	732
---	-----

Chad Coleman, Ryan Baker and Shonte Stephenson

Measuring Item Teaching Value in an Online Learning Environment	738
---	-----

Jon Harmon and Rasil Warnakulasooriya

Detecting Outlier Behaviors in Student Progress Trajectories Using a Repeated Fuzzy Clustering Approach.....	742
--	-----

Colm Howlin and Charles Dziuban

Content-based Course Recommender System for Liberal Arts Education	748
--	-----

Raphael Morsomme and Sofia Vazquez Alferez

Using a Glicko-based Algorithm to Measure In-Course Learning.....	754
---	-----

Rachel Reddick

Affect detection in home-based educational software for young children.....	760
---	-----

Roger Smeets, Francette Broekman and Eric Bouwers

Workshop Abstracts

Reinforcement Learning for Educational Data Mining	766
<i>Hamoon Azizsoltani, Min Chi, Joseph Jay Williams, Tiffany Barnes, Markel Sanz Ausin, Yeo Jin Kim and Anna Rafferty</i>	
Second LABBEC workshop: Learning analytics, Building bridges between the Education and the Computing communities}	768
<i>Sébastien Béland, Iris Bourgault Bouthillier, Michel Desmarais, Guillaume Loignon and Nathalie Loye</i>	
Designing and Developing Open, Pedagogically-Based Predictive Models using the Moodle Analytics API	770
<i>Elizabeth Dalton and David Monllaó Olivé</i>	
Sharing and Reusing Data and Analytic Methods with LearnSphere	773
<i>Kenneth Koedinger, John Stamper, Paulo Carvalho, Philip Pavlik and Luke Eglington</i>	
EDM & Games: Leveling Up Engaged Learning with Data-Rich Analytics	775
<i>Jonathan Rowe, Bradford Mott, Luc Paquette and Seung Lee</i>	
Causal Discovery with Tetrad in LearnSphere’s Tigris	777
<i>Richard Scheines, Peter Schaldenbrand and Kenneth Koedinger</i>	

On the Ethics of Data-Powered Instruction, Recommendation, Persuasion, and Behaviour Change

Keynote

Julita Vassileva
Department of Computer Science
University of Saskatchewan
Saskatoon, Canada
jiv@cs.usask.ca

Abstract

Advances in information technology and miniaturization, along with falling prices, allow affordable and powerful computing devices and sensors to become ubiquitous. These devices, equipped with advanced software and learning technologies resulting from 40 years of research in the areas of AI in education and cognitive modeling, can transform the way people learn and truly democratize education. Technological learning tools also generate valuable data about human learning processes and interactions. Through learning analytics and data mining, predictive models can be built enabling tailored or personalized support for individual learners and groups. For example, recent advances in reinforcement learning or partially observable Markov decision processes combined with semantic domain and pedagogical knowledge representations enable the design of optimized subject curriculums as well as flexible individualized learning and evaluation sequences. Progress in data mining including new fast algorithms for tensor decomposition enable recommending educational content appropriate to the learner's context, goals, knowledge, cultural background, personality, and motivational and affective states.

In parallel with developments in AI and Cognitive Psychology, research focus in the areas of Artificial Intelligence in Education (AIED) and Educational Data Mining (EDM) has broadened to address affective and social aspects of learning, as well as increasing the motivation of learners and engaging them in productive behaviours. Persuasive technologies, based on methods and strategies developed by neuroscientists, social psychologists and behavioural economists have shown effectiveness in engaging people in the desired behaviours.

This keynote talk focuses on the potential of persuasive technologies for supporting human learning. Many researchers in the AIED and EDM communities may not be aware of this potential. Persuasive technologies can make a bigger difference than gaining a fraction of percentage point on the RMSE of predicting the success rate over the next recommended item (a common debate on the value of accuracy of student skills modeling). Similarly, persuasive visualizations of individual or group student models facilitates processes of reflection, social learning, social comparison and competition, which have been shown to motivate learners and engage them in learning activities.

In my vision for the not-too-distant future personalized intelligent educational and persuasive technologies converge to support lifelong learning as people adapt to the rapidly changing natural and social environment. People will learn not only curriculum-based knowledge, but also knowledge on demand to achieve important goals like preserving the environment and enhancing civil society. Technologies of this scope face ethical challenges, related to their fairness and trustworthiness, power-balance, and the need to sustain the mental-health of people increasingly interacting exclusively with and through technology. Research attention in the Machine Learning (ML), Recommender Systems (RecSys), and Artificial Intelligence (AI) communities has already zoomed in on the ethical issues of the technologies they develop, specifically on the fairness, accountability and transparency (FAT) of data-driven systems, but understandably seeking mostly algorithmic solutions. Persuasive technology can provide methods for increasing the transparency of algorithms and correspondingly, the user trust in systems' data-driven decisions. Yet many of the problems with fairness and accountability are not algorithmic, but come from the complex, messy,

Julita Vassileva "On the Ethics of Data-Powered Instruction, Recommendation, Persuasion, and Behaviour Change" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 1 - 2

and inconsistent data fed to the algorithms. Bias in the ways data is collected, managed, annotated and used combined with lacking legal frameworks can lead to poor decisions, discrimination, violating the privacy of individuals, and exposing them to manipulation, which can have detrimental effects on society.

Data-hungry algorithms require big data, which leads to data hoarding. This generates multi-billion dollar businesses, and creates a great asymmetry in power between individuals, societies, or states and data-harvesting companies. Big data is opaque due to limited human cognitive processing power and attention span; full transparency seems intractable. A lot more research is needed to enable environments that offer equally good services, but are not so data-hungry, environments using small data, collected and used in context and then forgotten, managed by enforceable contracts with the users who can also gain rewards from sharing their data.

Personalized and persuasive social media applications has been blamed for being addictive by exploiting dopamine reward loops in the brain. We notice increased frequency of mental health issues, such as anxiety, depression, manias, addictions in young people but still we plan to engage them even more with our systems, reinforcing these problems. Maybe our systems should be teaching them how to detach from the technology, how to change their behaviours to engage with the real world and in real relationships, to meditate and find peace in the frenzy of information?

The keynote talk will address some of these issues and how persuasive technology, interactive visualization, and distributed ledgers can be used to provide solutions. It will also touch on research that isn't yet happening, but is needed, for example, enhancing learners' awareness of the dangers of personalization technology through personalized persuasive technology.

Bio

Dr. Julita Vassileva is a professor at the University of Saskatchewan and a leading researcher in user modelling, personalization and social computing, as well as AI in Education. She has authored over 250 research papers in the areas of intelligent tutoring and recommender systems, user modeling and personalization, trust and reputation, persuasion and behavior change. She is a member of the editorial boards of several reputed journals, including User Modeling and User Adapted Interaction, IEEE Transactions of Learning Technologies, ACM Transactions on Social Computing, International Journal of AI in Education and is the founding Editor-in-Chief of Frontiers in AI section on AI Supporting Human Learning and Behaviour Change. Dr. Vassileva was elected in 2017 to the Board of CS-Can |Info-Can (the Canadian Computer Science Society), and serves as a Chair of the Research Committee of the Board. Over her 20 years on faculty, she has advised over 35 graduate students (10 PhD and 27 MSc) and was nominated by her students and received the two main awards of the University of Saskatchewan for graduate supervision: the University's Distinguished Graduate Supervisor Award (2014), and the Advising Excellence Award by the Graduate Student Association (2013). Dr. Vassileva held the NSERC/Cameco Chair for Women in Science and Engineering for the Prairies region (2005 – 2011) and started a science outreach program for aboriginal youth in the North, called "Science Ambassadors", which has now completed 12 years of impactful activity in over 25 communities and has engaged over 25,000 indigenous youth in science. For her leadership and mentorship she was awarded the Saskatoon's YWCA "Women of Distinction Award (Science and Research)" in 2015.

Reconsidering two sigma: Educational data mining for the complete instructional system

Keynote

Steve Ritter
Carnegie Learning
Pittsburgh, Pennsylvania, USA
sritter@carnegielearning.com

Abstract

Educational software developers have long focused on achieving Bloom's (1984) goal of producing software that is as effective as a personal human tutor. Much progress has been made and, by some measures, this goal has been achieved (van Lehn, 2011). But maybe this is the wrong goal. The *two sigma* problem sets up a competition between teacher-based instruction and educational software. In the classrooms where Carnegie Learning's MATHia software is most used, there is no need to pick a winner. Most student usage of the software is in the presence of a teacher, and students can benefit from both. A better goal than improving on the teacher's instruction, then, may be to think of the teacher and the software as cooperating instructional resources with differing characteristics.

This perspective on the goal of educational software can lead to better design decisions focused on improving the overall educational experience for the student. Instead of attempting to reduce the teacher's classroom role, we focus on focusing the teacher on opportunities to have the biggest impact. During class, the software and the teacher communicate insights about student abilities and needs, enabling each to do a better job. In this talk I will illustrate ways that we use data to provide teachers with insights that help them assist students using MATHia. The teacher, in turn, provides MATHia with information that enables the software to perform better. While this is a new perspective for us, we are starting to see how this systems approach has led to even stronger educational outcomes for students.

Bio

Steve Ritter is Founder and Chief Scientist at Carnegie Learning. He has been developing and evaluating educational systems for over 20 years. He earned his Ph.D. in Cognitive Psychology at Carnegie Mellon University and helped start Carnegie Learning to ensure that the successful Cognitive Tutor approach was available to students across the country. Carnegie Learning's blended mathematics curricula, including MATHia software (formerly called Cognitive Tutor) are currently used by over 500,000 students each year.

Dr. Ritter was instrumental in the development and evaluation of the Cognitive Tutors for mathematics. He is the author of numerous papers on the design, architecture and evaluation of Intelligent Tutoring Systems and other advanced educational technology and is recognized as an expert on the design and evaluation of educational technology and on educational analytics. He is lead author of an evaluation that is one of the few to be judged by the US Department of Education's What Works Clearinghouse as meeting their standards without reservations. He has received several awards, including the Best Paper award at the International Conference on Educational Data Mining.

Dr. Ritter leads Carnegie Learning's research group, which is charged with focusing on improving student outcomes through data mining and field experimentation.

Steve Ritter "Reconsidering two sigma: Educational data mining for the complete instructional system" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 3

Educating a synthetic student

Keynote

Michael C. Mozer
University of Colorado at Boulder
and Google Brain
Boulder, Colorado, U.S.A.
mozer@colorado.edu

Abstract

In educational data mining, we use behavioral data to better understand learners and the nature of learning. I hope you agree with this characterization because it says nothing about whether the learner is a biological or a synthetic agent. In this talk, I focus on deep neural networks as learners. Modern nets have become so complex that we need to use experimental and modeling tools to characterize their behavior, just as we do for humans. Such analyses support the design of robust AI agents, but they also help to identify human-surrogate models that can be used to optimize instructional methods for people.

Bio

Michael Mozer received a Ph.D. in Cognitive Science at the University of California at San Diego in 1987. Following a postdoctoral fellowship with Geoffrey Hinton at the University of Toronto, he joined the faculty at the University of Colorado at Boulder and is presently a Professor in the Department of Computer Science and the Institute of Cognitive Science, as well as a Visiting Faculty Researcher at Google Brain (Mountain View). He is secretary of the Neural Information Processing Systems (NeurIPS) Foundation, has served as Program Chair and General Chair at NeurIPS and as chair of the Cognitive Science Society. He is interested in human-centric artificial intelligence, which involves designing machine learning methods that leverage insights from human cognition, and building software tools to optimize human performance using machine learning methods.

Michael Mozer "Educating a synthetic student" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 4

Predictiveness of Prior Failures is Improved by Incorporating Trial Duration.

Luke G. Eglington
Institute for Intelligent Systems
University of Memphis
lggln@memphis.edu

Philip I. Pavlik Jr
Institute for Intelligent Systems
University of Memphis
ppavlik@umemphis.edu

Abstract

In recent years, there has been a proliferation of adaptive learner models that seek to predict student correctness. Improvements on earlier models have shown that separate predictors for prior successes, failures, and recent performance further improve fit while remaining interpretable. However, students that engage in "gaming" or other off-task behaviors may reduce the predictiveness of learner models that treat counts of prior performance equivalently across gaming and non-gaming student populations. The present research evaluated how sub-groups of students that varied in their potential gaming behavior were differently fit by a logistic learner model, and whether any observed differences between sub-groups could inspire the creation of new predictors that might improve model fit. Student data extracted from a college-level online learning application were clustered according to speed and accuracy using Gaussian mixture modelling. Distinct clusters were found, with similar cluster patterns detected in three separate datasets. Subsequently, each cluster was separately fit to a Performance Factors Analysis model (PFA). Significantly different parameter coefficients across clusters implied that students more likely to have been gaming benefitted less from prior failures. These differences inspired new and modified predictors that were found to improve overall model fit - an improvement that varied in magnitude across clusters. The present findings indicate that incorporating trial duration into counts of prior failures can improve the predictive power of learning models.

Citation

Full article published as: Luke G. Eglington and Philip I. Pavlik Jr. Predictiveness of prior failures is improved by incorporating trial duration. *Journal of Educational Data Mining*, 2019

Luke G. Eglington and Philip I. Pavlik Jr. "Predictiveness of Prior Failures is Improved by Incorporating Trial Duration" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 5

Will this Course Increase or Decrease Your GPA? Towards Grade-aware Course Recommendation

Sara Morsy
Department of Computer Science and
Engineering
University of Minnesota, Twin Cities
USA
morsy@cs.umn.edu

George Karypis
Department of Computer Science and
Engineering
University of Minnesota, Twin Cities
USA
karypis@cs.umn.edu

Abstract

In order to help undergraduate students towards successfully completing their degrees, developing tools that can assist students during the course selection process is a significant task in the education domain. The optimal set of courses for each student should include courses that help him/her graduate in a timely fashion and for which he/she is well-prepared for so as to get a good grade in. To this end, we propose two different *grade-aware course recommendation* approaches to recommend to each student his/her optimal set of courses. The first approach ranks the courses by using an objective function that differentiates between courses that are expected to increase or decrease a student's GPA. The second approach combines the grades predicted by grade prediction methods with the rankings produced by course recommendation methods to improve the final course rankings. To obtain the course rankings in both approaches, we adapt two widely-used representation learning techniques to learn the optimal temporal ordering between courses. Our experiments on a large dataset obtained from the University of Minnesota that includes students from 23 different majors show that the grade-aware course recommendation methods can do better on recommending more courses in which the students are expected to perform well and recommending fewer courses in which they are expected not to perform well in than grade-unaware course recommendation methods.

Citation

Full article published as: Sara Morsy and George Karypis. Will this course increase or decrease your gpa? towards grade-aware course recommendation. *Journal of Educational Data Mining*, 2019. (to appear)

Sara Morsy and George Karypis "Will this Course Increase or Decrease Your GPA? Towards Grade-aware Course Recommendation" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 6

The Continuous Hint Factory - Providing Hints in Vast and Sparsely Populated Edit Distance Spaces

Benjamin Paaßen
CITEC Center of Excellence
Inspiration 1
33619 Bielefeld, Germany
bpaassen@techfak.uni-
bielefeld.de

Barbara Hammer
CITEC Center of Excellence
Inspiration 1
33619 Bielefeld, Germany
bhammer@techfak.uni-
bielefeld.de

Thomas William Price
North Carolina State
University
Campus Box 8206
Raleigh, NC 27695-8206, USA
twprice@ncsu.edu

Tiffany Barnes
North Carolina State
University
Campus Box 8206
Raleigh, NC 27695-8206, USA
tmbarnes@ncsu.edu

Sebastian Gross
Humboldt University of Berlin
Unter den Linden 6
10099 Berlin, Germany
sebastian.gross@informatik.
hu-berlin.de

Niels Pinkwart
Humboldt University of Berlin
Unter den Linden 6
10099 Berlin, Germany
niels.pinkwart@hu-
berlin.de

Abstract

Intelligent tutoring systems can support students in solving multi-step tasks by providing hints regarding what to do next. However, engineering such next-step hints manually or via an expert model becomes infeasible if the space of possible states is too large. Therefore, several approaches have emerged to infer next-step hints automatically, relying on past students' data. In particular, the Hint Factory (Barnes and Stamper, 2008) recommends edits that are most likely to guide students from their current state towards a correct solution, based on what successful students in the past have done in the same situation. Still, the Hint Factory relies on student data being available for any state a student might visit while solving the task, which is not the case for some learning tasks, such as open-ended programming tasks. In this contribution we provide a mathematical framework for edit-based hint policies and, based on this theory, propose a novel hint policy to provide edit hints in vast and sparsely populated state spaces. In particular, we extend the Hint Factory by considering data of past students in all states which are similar to the student's current state and creating hints approximating the weighted average of all these reference states. Because the space of possible weighted averages is continuous, we call this approach the Continuous Hint Factory. In our experimental evaluation, we demonstrate that the Continuous Hint Factory can predict more accurately what capable students would do compared to existing prediction schemes on two learning tasks, especially in an open-ended programming task, and that the Continuous Hint Factory is comparable to existing hint policies at reproducing tutor hints on a simple UML diagram task.

Citation

Full article published as: Benjamin Paaßen, Barbara Hammer, Thomas Price, Tiffany Barnes, Sebastian Gross, and Niels Pinkwart. The continuous hint factory - providing hints in vast and sparsely populated edit distance spaces. *Journal of Educational Datamining*, 10(1):1–35, 2018.

Benjamin Paaßen, Barbara Hammer, Thomas Price, Tiffany Barnes, Sebastian Gross and Niels Pinkwart "The Continuous Hint Factory - Providing Hints in Vast and Sparsely Populated Edit Distance Spaces" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 7

Using Sequence Mining to Analyze Metacognitive Monitoring and Scientific Inquiry based on Levels of Efficiency and Emotions during Game-Based Learning.

Michelle Taub and Roger Azevedo
University of Central Florida
12494 University Blvd
Orlando, Florida
{michelle.taub; roger.azevedo}@ucf.edu

Abstract

Self-regulated learning conducted through metacognitive monitoring and scientific inquiry can be influenced by many factors, such as emotions and motivation, and are necessary skills needed to engage in efficient hypothesis testing during game-based learning. Although many studies have investigated metacognitive monitoring and scientific inquiry skills during game-based learning, few studies have investigated how the sequence of behaviors involved during hypothesis testing with game-based learning differ based on both efficiency level and emotions during gameplay. For this study, we analyzed 59 undergraduate students' (59% female) metacognitive monitoring and hypothesis testing behavior during learning and gameplay with CRYSTAL ISLAND, a game-based learning environment that teaches students about microbiology. Specifically, we used sequential pattern mining and differential sequence mining to determine if there were sequences of hypothesis testing behaviors and to determine if the frequencies of occurrence of these sequences differed between high or low levels of efficiency at finishing the game and high or low levels of facial expressions of emotions during gameplay. Results revealed that students with low levels of efficiency and high levels of facial expressions of emotions had the most sequences of testing behaviors overall, specifically engaging in more sequences that were indicative of less strategic hypothesis testing behavior than the other students, where students who were more efficient with both levels of emotions demonstrated strategic testing behavior. These results have implications for the strengths of using educational data mining techniques for determining the processes underlying patterns of engaging in self-regulated learning conducted through hypothesis testing as they unfold over time; for training students on how to engage in the self-regulation, scientific inquiry, and emotion regulation processes that can result in efficient gameplay; and for developing adaptive game-based learning environments that foster effective and efficient self-regulation and scientific inquiry during learning.

Citation

Full article published as: Taub, M., & Azevedo, R. (2018). Using sequence mining to analyze metacognitive monitoring and scientific inquiry based on levels of efficiency and emotional expressivity during game-based learning. *Journal of Educational Data Mining*, 10, 1-26.

Michelle Taub and Roger Azevedo "Using Sequence Mining to Analyze Metacognitive Monitoring and Scientific Inquiry based on Levels of Efficiency and Emotions during Game-Based Learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 8

Mining University Registrar Records to Predict First-Year Undergraduate Attrition

Lovenoor Aulck
University of Washington
laulck@uw.edu

Dev Nambi
F.H. Cancer Research Center
dnambi@fredhutch.org

Nishant Velagapudi
University of Cal-Berkeley
nishray@berkeley.edu

Joshua Blumenstock
University of Cal-Berkeley
jblumenstock@berkeley.edu

Jevin West
University of Washington
jevinw@uw.edu

ABSTRACT

Each year, roughly 30% of first-year students at US baccalaureate institutions do not return for their second year and billions of dollars are spent educating these students. Yet, little quantitative research has analyzed the causes and possible remedies for student attrition. What's more, most of the previous attempts to model attrition at traditional campuses using machine learning have focused on small, homogeneous groups of students. In this work, we model student attrition using a dataset that is composed almost exclusively of information routinely collected for record-keeping at a large, public US university. By examining the entirety of the university's student body and not a subset thereof, we use one of the largest known datasets for examining attrition at a public US university ($N = 66,060$). Our results show that students' second year re-enrollment and eventual graduation can be accurately predicted based on a single year of data (AUROCs = 0.887 and 0.811, respectively). We find that demographic data (such as race, gender, etc.) and pre-admission data (such as high school academics, entrance exam scores, etc.) - upon which most admissions processes are predicated - are not nearly as useful as early college performance/transcript data for these predictions. These results highlight the potential for data mining to impact student retention and success at traditional campuses.

1. INTRODUCTION

Student attrition has long been a topic of great interest in higher education research, with government reports on attrition dating back over 100 years [31]. This interest stems from the fact that students who do not graduate are a lost investment on many fronts. For higher education institutions, limiting attrition is central to their financial sustainability as they devote scarce resources towards classes and services for non-completing students [17]. In particular, it is estimated that 30% of United States (US) first-year students do not return for their second year of post-secondary education with

US taxpayers spending nearly \$2 billion annually on educating non-returning first-year students alone [28]. Institutions are also concerned with attrition rates because they are central to estimates of institutional effectiveness, thereby affecting funding opportunities and government support [14]. Highlighting the impact of attrition at the institutional level also says nothing of its impact on students, who devote time, effort, and finances towards unfinished educational pursuits. Leaving college drastically alters career trajectories for students and those without college degrees face continually declining job growth and worsening job prospects [9].

In light of this, understanding motivations for students to drop out and possible remedies thereof is of great importance [12]. Empirical evidence to build student attrition theory has traditionally focused on survey-based research [30, 8]. However, survey instruments are often costly to implement, time-consuming for data collection, and produce results that are not always generalizable across institutions due to vastly different student profiles [34, 7, 8]. Institutional data that is routinely collected at colleges and universities (e.g. student application and transcript data) can provide an alternative data source and a way to supplement survey-based measures [8]. Leveraging data sources already in existence can add a means to more efficiently examine the student attrition problem and help institutions remedy the issue of attrition. One field that is primed to take advantage of this institutional data is educational data mining (EDM) and its focus on data-intensive techniques in educational settings [26, 4].

EDM is an emerging field with much of its research on attrition centered on massive online open courses (MOOCs) and other online environments (e.g. [35, 13]). Studying attrition in MOOCs and other online settings lends itself to expansive data collection opportunities and a detailed monitoring of students [23]. This limits the extent to which this work can be generalized to more traditional campus settings (i.e. campuses where learning is primarily on-campus, in-classroom). Meanwhile, EDM-centric work on predicting attrition at traditional campuses has been scarce and usually limited to small, homogeneous subsets of students rather than the entirety of a college student population. Additionally, the focus when predicting attrition is usually on how well it can be predicted and less so on what type of data is best for these predictions.

In this work, we predict the attrition of a large number of un-

Lovenoor Aulck, Dev Nambi, Nishant Velagapudi, Joshua Blumenstock and Jevin West "Mining University Registrar Records to Predict First-Year Undergraduate Attrition" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 9 - 18

dergraduate students ($N = 66,060$) using only their first year of academic data. The students we examine are not from a single department or major within a university. Rather, they span the entirety of a student body, thereby comprising a dataset with heterogeneous aspirations, backgrounds, and goals. In addition, we rely almost entirely on data that is routinely collected at institutions of higher education. With this data, we seek to answer two questions: to what extent can undergraduate student attrition be predicted using a limited amount of data from registrar records and what types of data from registrar records are most useful in predicting attrition. The first of these has been explored in the past while using smaller and/or homogeneous student populations; the second has not been systematically examined in the literature to our knowledge.

To answer the above questions, we mine the institutional data records at a large, public university in the US and engineer features for predictions. We then create numerous machine learning models using the engineered features and compare the performance of these models to each other. Then, we create separate machine learning models using only groups of features and not the entirety of the feature space to compare the predictive power of different subsets of institutional data. This work is an extension of our previous work on modeling student attrition using a limited amount of data [3] but where we previously focused on using the *first term's* data in generating features for prediction, we use the *first year's* in this work. We also extend our previous work to build additional machine learning models, predict attrition as defined according to two different definitions (overall graduation and re-enrollment after students' first year), and examine the types of feature subsets most useful in predictions. In so doing, we present two key findings, both of which have many implications for administrative policy in higher education:

- We demonstrate that the graduation and second-year re-enrollment of students can be predicted using data that is routinely gathered at institutions of higher education.
- We show that demographic and pre-entry features have less predictive power than data on student academics.

2. RELATED WORK

There are many examples of predicting attrition at traditional campuses. Most of these focus on small, homogeneous subsets of students. Moseley predicted the graduation of 528 nursing students using rule induction methods, obtaining high accuracies but not controlling for the number of terms/semesters examined for each student [21]. Dekker et al. looked at only the first semester grades of 648 students in the Electrical Engineering department at the Eindhoven University of Technology and were able to predict dropout with 75-80% accuracy [10]. Kovačić used tree-based methods on a similarly-sized dataset of 453 students at the Open Polytechnic of New Zealand, finding ethnicity and students' course taking patterns to be highly useful in prediction [18]. Bayer et al. looked at 775 applied informatics students at the Czech Republic's Masaryk University across three years [5]. Without limiting the amount of information available for each student, they found that including features related to students' social behavior can boost prediction accuracy by over 10% for some models. These and similar studies, how-

ever, focus on relatively small (e.g. $N < 2,000$) subgroups of students with similar academic pursuits/foci. In addition, there is little consistency with respect to the timeframes across which data is examined for each student. Other approaches to predict attrition at traditional campuses include early alert systems, which are often labor intensive and poorly funded [29]. These alert systems have been shown to positively benefit students (e.g. [16]), but usually rely on data gathered in the midst of a course or an academic term (e.g. [27, 15]), which may not always be feasible.

The work we present more closely relates to a subset of literature looking at student attrition in the context of the heterogeneity of students across an entire campus and not just a subset thereof. Our work also deals with much larger student populations than those described above and, in this sense, it more closely resembles a more recent body of literature. Delen used 8 years of institutional data on over 25,000 students at a large, public US university, predicting whether the students would return for their second year [11]. However, due to class imbalances, Delen re-sampled the majority class and ultimately used only 6,454 students for predictions. Ram et al. used data on about 6,500 freshmen at a large, public US university to predict whether students would drop out after their first semester, and for those that did not, whether they will drop out after an additional term [25]. Ram et al. supplemented data from institutional databases with student smart card transactions to infer social integration. More recently, Nagy and Molontay predicted the dropout of 15,825 students from the Budapest University of Technology and Economics using only their information prior to college entry with some success [22].

There are a few ways in which our work contributes to this body of literature. Firstly, we use a much larger dataset than has been previously examined specifically for attrition (66,060 students). We examine the entirety of a large university's student body and we do not limit the extent of heterogeneity of the students in the dataset. Additionally, we also address the question of what types of features are most useful in predicting student attrition. In particular, previous works have generally used all available data sources concurrently in determining which students will attrite. In this work, we explore what types of routinely-collected institutional data fare best when predicting attrition by comparing performance using different data subsets in isolation. Finally, we concurrently compare predictions for two different definitions of "attrition," highlighting the degree to which operationalizing the term can impact results.

3. METHODS

We describe the methods for this work by first detailing the data used in the project. We then give relevant operational definitions with respect to how we define attrition. Thereafter, we discuss the data subsets used in the predictions and the features generated. Lastly, we describe the setup of the machine learning experiments.

3.1 Data Description

We collected pseudonymized, de-identified data from the University of Washington (the University) data stewards in 2017. The University is a traditional campus setting where a vast majority of instruction is in person and face-to-face. No

personally identifiable information was collected for the students; instead, students were referenced using unique identifying keys. Table 1 shows the tables that were pulled from the registrar databases. In general, the data included information on students’ demographics, complete transcript records at the University, and information from applications to the University. We did not have any information on students’ financial aid status or economic status other than that which was derived from their ZIP code, as described below. Socioeconomic factors can play a large role in the student attrition process [6], however, we did not have access to student finances for use in this work. We also did not have access to any exit surveys from students who had either left the University or had graduated.

Table 1: Data pulled from registrar databases

Table	Description
Application Data	Information from student applications to the University including high school coursework
Guardian Data	Information on student guardians as pulled from student applications to the University
Demographic Data	Information on student demographics including date of birth, race, ethnicity, gender, etc.
Major Data	Information on majors declared by students on a term-by-term (quarter-by-quarter) basis
Test Score Data	Information on student standardized test results
Transcript Data	Information on student coursework and grades on a term-by-term (quarter-by-quarter) basis

We restricted data to high school graduates who first enrolled at the University as matriculated, baccalaureate-degree-seeking undergraduate students between 1998 and 2010 without previously attending another post-secondary institution full-time. These students are henceforth referred to as “freshmen.” The dataset included students who were in a college in high school program but excluded those who attended junior/community college full-time after high school and then transferred to the University. Because the data was pulled in 2017, we used the year 2010 as a cutoff to allow for six full years of visibility on student academics at the University before labelling a student as a “non-completion,” as defined in Section 3.2. In total, the dataset consisted of 66,060 unique freshmen entrants. We then further limited the data for each student to information through one calendar year from each student’s first enrollment at the University. This data was limited to one calendar year for all students, regardless of the number of courses they took/passed, their grades, or their backgrounds.

After joining tables of interest using the unique student identifiers, we created features for the prediction experiments by either pulling them directly from the raw data or engineering them for each student. The features were grouped in 7

groupings, which are described in Section 3.3; a comprehensive list of features and descriptions thereof is available upon request but was not provided in this writing in the interest of space. In total, there were 1,405 features and all features were generated for each student without exception.

3.2 Definitions

Ambiguity with respect to operational definitions of dropout in literature on student attrition can make it difficult to compare results across studies [24, 33]. There are numerous ways in which attrition has been defined in existing literature, be it students dropping out from a particular course (e.g. [21]), re-enrolling after their first term (e.g. [1]), re-enrolling after their first year (e.g. [11]), graduating on time (e.g. [3]), or reaching some other relevant milestone (e.g. [10]). In this work, we defined attrition in two ways and analyze both. We examined attrition from students’ first year to their second (“re-enrollment” and “non-re-enrollment”) as well as looking at whether a student graduated on time (“graduate” and “non-completion”). We do not examine attrition on a term-by-term basis because of the relatively few students who leave the University after only a single term, as discussed in Section 4.1. We operationally defined non-completion and re-enrollment as described below.

3.2.1 Non-Completion

We defined “non-completion” as any freshman student who did not graduate with a baccalaureate degree from the University within 6 calendar years of first entry to the University. We defined a “graduate” as a freshman who graduated from the University with a baccalaureate degree within 6 calendar years of first enrollment. The University uses a quarter term system and we used the span of four consecutive academic quarters as a measure of one calendar year. Six calendar years for graduation was thus the span of 24 consecutive academic quarters. This definition of non-completion only accounted for students’ first baccalaureate degree and did not take into account double-majors or double degrees. For example, if a student was simultaneously pursuing two baccalaureate degrees but only graduated with one in five years, they would be a graduate; alternatively, if the student had graduated with both degrees but during their seventh year, they would be considered a non-completion. Because we focused on registrar records from a single institution, defining non-completion in this manner does not take into account students’ academic progression after leaving the University. This is because we only had access to registrar records from a single institution and did not track students across multiple institutions - they could have very well transferred from the University and graduated in good standing.

We accounted for students who took part in a college in high school program by converting their transferred credit total to a count of academic quarters completed while assuming typical full-time enrollment at the University. For example, if a student completed 30 credits in a college in high school program, we converted this credit total to a count of terms completed at the University (in this case, 2, as students typically take 15 credits per term). We rounded the result from this conversion where appropriate. We then deducted this number when determining whether the student had graduated within an appropriate amount of time.

3.2.2 Re-Enrollment

We defined “re-enrollment” as a student who completed at least one additional course within one calendar year of the end of their first calendar year at the University (i.e. within 4 academic quarters from the end of their first year). “Non-re-enrollments” were students who were not re-enrollments. In this work, the definitions of graduation and re-enrollment were treated mutually exclusive in that all graduates were not necessarily re-enrollments. It should be noted that the University requires students who do not enroll for two consecutive terms without an excused leave to be re-admitted at the discretion of the University.

3.3 Feature Groupings

For every student, we engineered the subsets of features that are described below. For all student grades, we calculated a grade percentile and a z-score by comparing each students’ grades to the grades of all undergraduate students who had taken the same course at the same time. References to grades include the student’s GPA (on a 4.0 scale), their percentile score (from 0-100), and their z-score for courses (representing the number of standard deviations from the mean, assuming a normal grade distribution). References to “performance” for the feature groupings include grades and credits earned, at the least. In some cases, references to performance may also include the number of graded credits earned (versus courses taken pass-fail) and the number of credits attempted. A brief description of each of the feature subsets is provided in Table 2.

Table 2: Data subsets used in predictions

Subset	Description
Base Data	Year and quarter of University entry (included with every other data subset)
Demographic Data	Non-academic data prior to entry to the University, including demographics
Department-level Data	Measures of performance aggregated by course department
First-Year Summary Data	Aggregated measures of academic performance during first year
Grouped Course Data	Measures of performance aggregated by course number and STEM gatekeepers
Major Data	Counts of majors declared on a term-by-term basis
Pre-Entry Data	Academic data prior to entry to the University.

3.3.1 Base Data

Base data consisted of only three features and was included in the feature space when making predictions using every other data subset described. The base data included students’ calendar year of entry to the University, their quarter of entry to the university (i.e. which of the four academic quarters was a student’s first; ranging from 1 to 4, with 1, 2, 3, and 4 corresponding to winter, spring, summer, and autumn academic quarters, respectively), and a quarter-year variable which consisted of students’ year of entry multiplied by 4 and added to the quarter of entry to create a relative

time scale. These features were included to account for any time-related variation in graduation rates.

3.3.2 Demographic Data

Demographic data consisted of student’s non-academic information prior to entry to the University. This included, but was not limited to, students’ gender, race, ethnicity, age at college enrollment, veteran status, and student athlete status. We also included information from students’ application to the University, such as information on the students’ high schools (excluding high school grades), parents’ educational attainment, and students’ ZIP (postal) code, which was either pulled from their high school information or, when unavailable, from their university application. We joined students’ ZIP codes with 2015 US census data¹ to find the average income and educational attainment in each ZIP code. We also included the distance from the University to each student’s home ZIP code. Features derived from ZIP codes were the only features from sources external to the University’s registrar databases.

3.3.3 Department-level Data

Department-level data consisted of student performance in course offerings grouped by course prefix. For example, this included performance in all BIOL (biology) courses grouped together, performance in all HIST (history) courses grouped together, etc. We excluded course prefixes wherein at least 10 students from the dataset did not take a course. In all, this included 200 unique course prefixes and 1000 features, with GPA, percentile grade, z-score, credits earned, and graded credits earned calculated for each prefix. We used department-level data instead of individual course data after preliminary modeling using individual courses did not yield strong results. The expansive feature space when engineering features across individual courses also significantly increased the requisite computational power/time for modeling and we decided against pursuing this further.

3.3.4 First-Year Summary Data

First-year summary data consisted of aggregate measures of students’ first year at the University. This included, among other things, students’ course performance, credits taken, number of courses failed, number of quarters enrolled, and enrollment in a freshman seminar courses. The first-year summary data also included aggregate measures of students’ performance in their first, second, third, and fourth quarters as well as student performance in the last academic quarter for which they were enrolled during their first year (regardless of which quarter it was). We also included differences between students’ performance in successive quarters.

3.3.5 Grouped Course Data

Grouped course data consisted of student course performance grouped either by course number or by performance in “STEM gatekeepers.” To group courses by course number, we aggregated performance across all courses that were numbered below 100, from 100-199, from 200-299, from 300-399, and 400+. The course numbering generally reflected whether the course was designed to be taken by lowerclassmen or upperclassmen and, in some cases, also indicated

¹From the US Census Bureau’s American Fact Finder

during which year students typically took the course. STEM gatekeepers refer to introductory science, technology, engineering, and math (STEM) courses which often function as pre-requisites for STEM majors and degrees. These gatekeeper courses tend to be highly competitive and performance in these courses is a key determinant of whether a student will be accepted into any of the highly competitive STEM majors. We grouped the performance in STEM gatekeepers by course department and topic (e.g. the calculus series, the general chemistry series, the organic chemistry series, etc.) as well as across all STEM gatekeepers.

3.3.6 Major Data

Major data consisted of counts of students' major declarations during their first academic year. In most cases, students entered the University with a "pre-major" designation before declaring their major(s) of interest some time during their first or second year. These pre-major designations varied based on field of interest (e.g. pre-engineering, pre-nursing, pre-health, etc.). Students' majors were recorded on a per-quarter basis by the University (once per quarterly transcript record) and we tallied the counts of major declarations for each student across the entirety of their first year. For example, a student who declared a math major in their first two quarters only to switch to geography in their third quarter and then add a history double major in their fourth quarter would have the values 2, 2, 1 in the math major, geography major, and history major features, respectively.

3.3.7 Pre-Entry Data

Pre-entry data consisted of students' academic information prior to attending the University. This included, among other things, students' entrance exam scores, high school GPA, high school coursework, and college in high school program participation and performance. We did not include any information on students after their enrollment at the University in the pre-entry data.

3.4 Machine Learning and Predictions

We randomly divided the students into training and test sets using a 80-20 split (N in training = 52,848; N in test = 13,212). We used the same test set when evaluating the predictive performance of each of the models to allow for direct comparisons to be made. The data was highly skewed with graduates and re-enrollments comprising 78.5% and 93.1% of all the data, respectively. Graduates and re-enrollments comprised 78.0% and 92.9% of the test data, respectively. Though dealing with class imbalances is of great interest when examining freshmen attrition [32], we did not use any balancing techniques as we wanted to work with the data in its original, unaltered form. We scaled the training data by subtracting the median of each feature and dividing by the respective feature's interquartile range. We subsequently scaled the test data using the scaling values for each feature from the training data.

We used five different machine learning models to predict each student's graduation and re-enrollment: regularized logistic regression (LR), K-Nearest Neighbors (KNN), random forests (RF), support vector machines (SVM), and gradient boosted trees (XGB). We trained each model across the entirety of the training data and used the same training

instances to train each of the models. We trained each model separately to predict graduation and re-enrollment. We tuned model hyperparameters for each model using 5-fold cross validation on the training data, after which the models were re-trained on the entirety of the training data using the tuned hyperparameters. We report final error metrics and performance on the test set, which was consistent across all models, regardless of whether predicting graduation or re-enrollment.

After developing predictive models using all features, we created regularized logistic regression models using each of the 6 feature subsets highlighted in Section 3.3 in isolation. The base data (see Table 2) was included in the feature space for each data subset. The rationale behind using regularized logistic regression for these models is further discussed in Section 4.3. We understand that an alternative approach would be to test all the models listed above for each of the data subsets to find the best performing model/subset combinations. That said, we believe our approach was still suitable for comparing different data subsets. When modeling using data subsets, we used the same observations as before to train each of the models and, as before, we developed a separate model for predicting graduation and re-enrollment for each of the data subsets. As such, the *training instances* were the same across models but the *training features* differed depending on the feature subset used. We tuned the regularization strength for these regularized logistic regression models using 5-fold cross validation on the training dataset and we report results on the test set.

4. RESULTS AND DISCUSSION

4.1 Student Characteristics

We show the number and proportion of graduates and re-enrollments in Figure 1. In all, 78.5% of students were labelled graduates while 93.1% of students were labelled re-enrollments. These proportions were verified with the University's office of institutional analysis. Such highly skewed data towards graduates and re-enrollments can be expected in a large, tier-1 research university setting where there has been considerable, long-standing effort to improve the overall attrition rate over time. That said, it must also be noted that at an institution with such a large student population, even small fractions of the student body represent hundreds of students on an annual basis. Across the timeline of the dataset (13 cohorts), 14,196 non-completions and 4,593 non-re-enrollments represent 1,092 and 351 students on an annual basis, respectively.

We show the cumulative percentage of students who either graduated or left the University across time in Figure 2. We used the first year as a cutoff for the data because, historically, a large number of students decide whether they will continue with their higher education pursuits during and immediately after their first year [28]. As such, developing models that can predict whether students will re-enroll for a second year and whether they are on a trajectory towards successful graduation could help administrators and academic advisors more effectively develop and deliver interventions directed towards students in need of assistance. When examining the data, 27.5% of all non-completions leave the university prior to the start of their 2nd year, 51.9% of non-completions leave the University between their 2nd and 6th

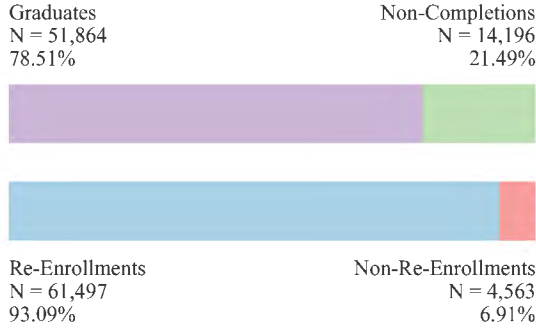


Figure 1: Counts and percentages of classes in the dataset. Definitions are provided in Section 3.2.

year, and 20.6% continued to be enrolled at the University after their 6th year. The difference in number between non-completions who did not return for their 2nd year and non-re-enrollments can be attributed to non-re-enrollments who later returned to the University and graduated on time. Less than 5% of non-completions and less than 15% of non-re-enrollments left the University after only one term, leading us to not examine attrition after the first and second terms. In settings where attrition rates are higher after students' first and second terms, it may be more relevant to examine the performance of classifiers after one or two terms.

Figure 2 also shows that a majority of graduates (65.6%) completed their degrees during their fourth year at the University. The mean and median completion time for all graduates was 16.6 and 15.0 calendar quarters, respectively, from first enrollment. This is particularly apparent due to the near-sigmoidal shape of the cumulative graph for graduates, with a sharp rise during students' fourth year. We also see that there is a relative lack of students who graduated prior to the start of their third year. This highlights the difficulty in predicting graduation based on students' first year - a student typically does not graduate until several years later, during which a host of influences can shape an academic trajectory, be they personal, financial, or academic.

4.2 Predictions Using Different Algorithms

Table 3: Prediction results using all data features. Baseline values are based on test set.

Model	Graduation		Re-Enrollment	
	Accuracy	AUROC	Accuracy	AUROC
Baseline	78.0%	0.500	92.9%	0.500
LR	83.2%	0.811	95.0%	0.882
RF	83.1%	0.806	95.3%	0.887
XGB	83.0%	0.806	95.1%	0.885
KNN	82.5%	0.798	94.8%	0.876
SVM	78.0%	0.780	92.9%	0.862

We show the performance of each of the models using the entirety of the feature space in Table 3. The baseline measure in the Table refers to the majority class compositions in the test set. Generally speaking, most of the models had a similar comparative performance for each prediction task (i.e. predicting either graduation or re-enrollment). This hints at

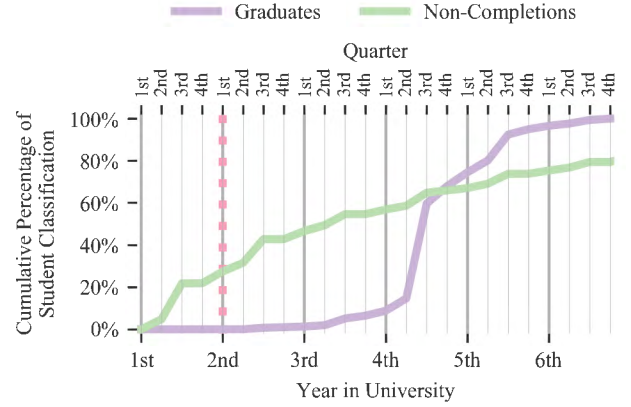


Figure 2: Cumulative graduation and non-completion curves of students. Years and quarters are relative to the time of first enrollment. The dotted line indicates the point to which data is limited for each student. Only students' first six years are shown, per the definition of "graduate."

an effective ceiling with respect to predictive power from the types of features being used (i.e. ones pulled from registrar records) and that additional representations of the student experience (be they academic or social) should be incorporated. Alternatively, a more complex predictive model (e.g. deep neural networks) may also fare better in making these predictions. That said, given the data used, the models are able to predict the eventual graduation and re-enrollment of students fairly successfully, as evidenced by the relative improvements over baseline values for both prediction tasks.

For predicting graduation, logistic regression was the best-performing model, followed by random forests. When predicting re-enrollment, random forests performed the best, followed by gradient boosted trees and logistic regression. These results are generally in line with our previous work on similar tasks, where we found that logistic regression tends to work well compared to other models for predicting graduation and STEM attrition [2]. When examining the worst-performing models, the SVM model made predictions that consisted entirely of the majority class when predicting both graduation and re-enrollment, as seen by the models' accuracy being the same as the baseline values. Such results are typical of classifiers without much predictive strength on a dataset consisting of highly disproportionate classes. In this specific case, it may be remedied by using alternate kernels for the model, which we did not explore in this work.

We show the ROC curves for the models in Figure 3. These curves further illustrate the lack of differentiation with respect to model performance. For the same prediction task, the resulting ROC curves across the models were nearly identical with little difference in curvature. The more notable difference was when comparing the ROC curves for predicting graduation with those for predicting re-enrollment, as the curves for predicting re-enrollment were more prominently convex compared to those for predicting graduation. These curvatures, along with the metrics shown

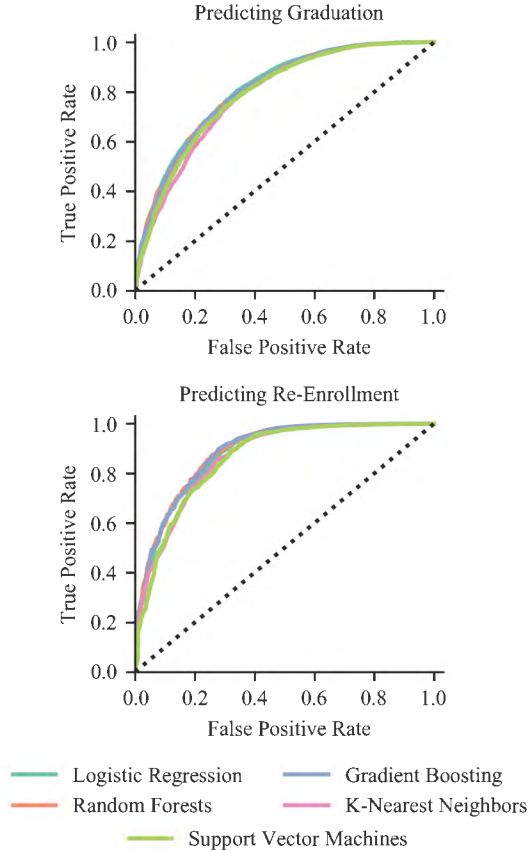


Figure 3: Receiver operating characteristic curves when using different machine learning models.

in Table 3, demonstrate that predicting students’ eventual graduation is a more difficult task than predicting students’ re-enrollment. We expected this as the cutoff for the data used in the predictions (i.e. students’ first year) was near the point at which a student is classified as a re-enrollment (after their second year) but was much earlier than when a student was classified as a non-completion (after their sixth year). This helps highlight the degree to which differing operational definitions of attrition can vastly alter the perceived predictive strength of these classifiers. For other scenarios, alternate definitions of attrition may be more appropriate and the effectiveness of efforts to build predictive models will be colored by these definitions and institutional contexts.

We show the confusion matrices for the best models for predicting graduation and re-enrollment (logistic regression and random forests, respectively) in Figure 4. These matrices show a lower rate of false negatives for the models but a higher rate of false positives (i.e. students incorrectly classified by the models as having graduated or re-enrolled). To better understand this higher rate of false positives, we examined the complete transcript records of students who were classified accordingly. Across the false positives, we found numerous instances of non-completions and non-re-enrollments who had left the University with relatively strong grades in comparison to their graduating and

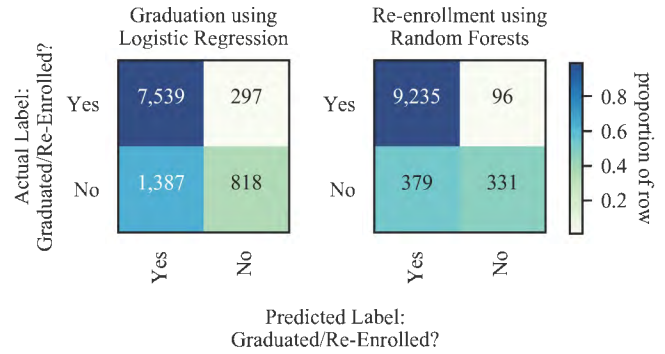


Figure 4: Confusion matrices when examining the top performing algorithms for predicting graduation (LR, left) and re-enrollment (RF, right).

re-enrolling peers. These students also often appeared to be pursuing very competitive majors and/or appeared to have rigorous post-graduation plans (e.g. pre-medical and pre-dental students). Many of these students remained in a pre-major state prior to their departure, indicating that though they had relatively strong grades, they likely were not able to enter into their degree program(s) of choice for various reasons and had to leave the University to pursue these ambitions as a result. Unfortunately, the University does not have a centralized major application database for admissions and rejections to specific majors. Having so could shed light on much of the motivation behind these students’ desire to leave the University and if it was, in fact, motivated by not getting into competitive majors. That said, the fact that many of these students were academically similar to their graduating and re-enrolling counterparts further illustrates why there appears to be an effective ceiling with respect to predictive power using the given data, as seen in Table 3.

From a practical perspective, it should be noted that the classification thresholds for these models were not tuned with respect to either sensitivity or specificity. In practice, when developing institutional systems to identify students at-risk of leaving, it may be useful to raise the classification threshold when predicting whether a student will graduate or re-enroll, thus favoring lower recall at the expense of higher precision. This would effectively reduce the number of students who are predicted to graduate but in actuality do not (i.e. false positives) at the expense of more false negatives, which could be more acceptable when developing an alert system for students at risk of dropping out.

4.3 Predictions Using Different Data Subsets

After examining the results from predicting graduates and re-enrollments using all features, we used regularized logistic regression to predict graduation and re-enrollment using subsets of the data. We used logistic regression after we saw that it performed very well relative to other models for both prediction tasks (see Section 4.2) and because it had relatively fast training times due to having fewer hyperparameters to tune. This allowed us to more efficiently train the 12 different models that were needed when examining the performance of specific data subsets (i.e. separately modeling graduation and re-enrollment while using 6 different

Table 4: Prediction results using specific data subsets. Baseline values are based on test set.

Subset	Graduation		Re-Enrollment	
	Accuracy	AUROC	Accuracy	AUROC
Baseline	78.0%	0.500	92.9%	0.500
All	83.2%	0.811	95.0%	0.882
FY-Sum.	83.0%	0.795	94.9%	0.855
Department	82.3%	0.788	94.6%	0.847
Grouped	82.5%	0.781	94.6%	0.845
Major	79.9%	0.661	94.2%	0.768
Demo	78.0%	0.634	92.9%	0.643
Pre-Entry	77.3%	0.630	92.9%	0.616

data subsets in isolation for each).

We show the results when using data subsets in Table 4 alongside the performance of the logistic regression classifier from Section 4.2. Transcript-based features tended to perform better than information on students’ prior to their enrollment at the University. More specifically, demographic data and pre-entry information did relatively poorly in predicting both graduation and re-enrollment. Intuitively, this is not a surprise as the admissions process at highly-competitive universities tends to be fairly selective with an emphasis on supporting and sustaining a successful yet diverse student body. Additionally, such institutions may already have efforts in place to reduce demographic disparities for student success. Meanwhile, when looking at transcript-based data subsets, first-year summary data performed the best with performance that was similar to using the entirety of the data. This is particularly noteworthy as the first-year summary data contained fewer features than the other transcript-based data subsets but was centered on summaries of performance across time rather than aggregations across course departments/numberings.

These findings are particularly interesting in light of work by other researchers. For instance, Nagy and Molontay found that attrition could be accurately predicted using what we outline as demographic and pre-entry features alone [22]. However, we do not see similar success here. We believe this could be due to vastly different educational settings and student profiles (e.g. here, most students tend to graduate/re-enroll while Nagy’s student population primarily dropped out). In earlier work, Dekker et al. found that transcript-based features tend to have more predictive strength than pre-entry features, but examined this across rather limited data subsets [10]. Our results echo this finding. Recently, Manrique et al. found that attrition could be predicted using student performance in a few key courses [20]. Here, we find that aggregates across the first year tend to work better than more fine-grain representations of course-taking (e.g. grouping classes by course prefix and numbering). As discussed in Section 3.3.3, we decided against using individual course representations in this work.

We show the ROC curves for the regularized logistic regression models using each of the data subsets as well as the entire feature space in Figure 5. The fact that demographic and pre-entry data gave generally worse performance than

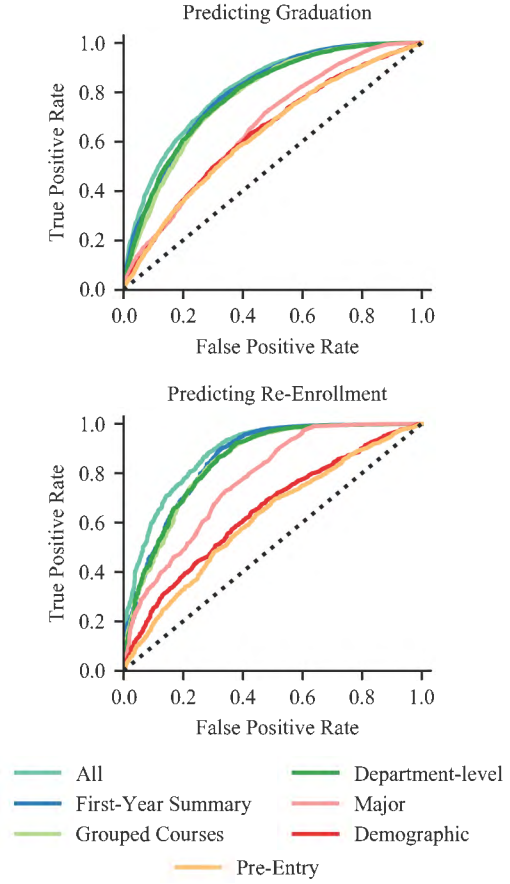


Figure 5: Receiver operating characteristic curves when using different subsets of data.

transcript-based features is very much apparent from the ROC curves. Data on majors, meanwhile, tended to perform worse than other transcript-based features but better than demographic and pre-entry data. The fact that using data on majors did not yield particularly strong results likely relates to the fact that most students in the dataset were in a pre-major state across their first year and formally declared their major of interest later in their undergraduate careers. As noted above, a centralized major application system was not available, else it could have been leveraged in addition to data on majors to draw a more clear picture of student academic interest. The other transcript-based datasets, meanwhile, had very similar curvatures for the ROC curves when predicting both graduation and re-enrollment.

We show confusion matrices from using the best-performing data subset in Figure 6. The best-performing data subset for both prediction tasks was first-year summary data. By comparing these confusion matrices to those shown in Figure 4, it can be seen that using just a limited subset of features tends to classify the data similarly to models built on the entirety of the data. This is true not only in terms of how effective the models are in making predictions, but also with respect to the relatively high rate of false positives seen across all four matrices.

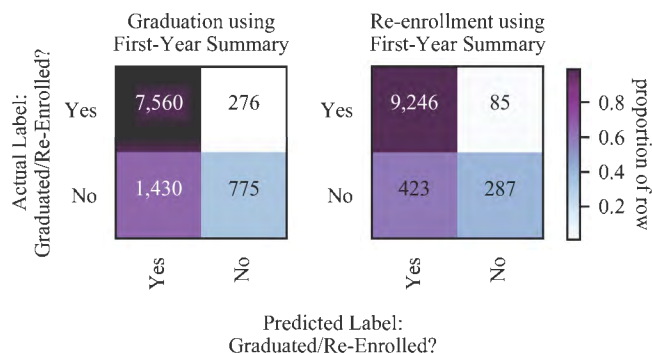


Figure 6: Confusion matrices when examining the top performing data subset for predicting graduation (left) and re-enrollment (right). The top performing data subset was the same for both tasks (first-year summary data).

5. FUTURE DIRECTIONS

We believe the findings regarding the data subsets have wide-ranging policy implications, particularly for identifying students at risk of dropping out in large, public universities. In such settings, there may be longstanding effort to decrease demographic disparities with respect to attrition and, as a result, transcript records may be more viable as features in predictive models than pre-entry/demographic information. Furthermore, these settings may also be resource-constrained with respect to time available for staff to hand engineer features. In such settings, knowing which features would be most predictive of attrition without the need to hand-engineer features across the entirety of data available to institutions could save time and effort in building models. We have had conversations with administrators at the University for better interpreting our results and improving the processes for identifying students in need of assistance.

Another direction of interest is better understanding the features used in predicting attrition. This includes not only further examining key individual determinants of attrition, as we have done in previous work [3, 2], but also finding the best combination of features across the subsets. We would like to examine this “minimum viable feature space” in the context of data available in registrar databases as well as investigate the degree to which these features relate to established theory on student attrition [12].

6. CONCLUSIONS

In this work, we use data from the registrar databases of a large, public US university to predict both graduation and re-enrollment using information limited to students’ first calendar year at the university. We do this using a dataset of students that spans the entirety of the university student body and is thus much larger than previous studies predicting student attrition ($N = 66,060$). In so doing, we demonstrate that both graduation and re-enrollment can be effectively predicted using features generated from data that is routinely collected at institutions of higher education. Additionally, we also examine the degree to which specific subsets of registrar data can be useful in predicting attrition, finding that transcript-based features tend to outperform features

based on student histories prior to college. This implies that effective strategies for intervention can be outlined based on registrar records.

Predicting re-enrollment after students’ first year was a much more tractable task than predicting graduation. This can be attributed to the fact that predicting graduation necessitates predicting academic success years into the future from the point to which data was limited whereas predicting re-enrollment is within a much shorter timeframe. Considering the unpredictable influences that cause students to leave college prior to graduating (e.g. financial limitations, personal hardships, etc.), a more reliable prediction task may be to examine whether a student will return on a term-by-term basis. This could be particularly useful to develop alert systems to identify students at risk of dropout. However, this was not explored in this work due to the relatively few students who left the University after a single term.

We found that there appears to be an upper limit for predictive power for our dataset. This demonstrates the limitations when relying solely on registrar data and shows the need for additional features on the student experience to improve predictive power. Some potential features of interest include measures of social integration on campus and of financial aid. Better understanding student aspirations beyond simply using declared majors could also be of interest, especially using alternate representations of student course-taking behavior, as shown recently by Luo and Pardos [19].

Lastly, we show that features generated from transcript records, particularly aggregates and summaries of students’ academics, perform better for predictions than demographic and pre-entry data. Much of this is likely due to the selectivity of the University and its admissions policy. Nevertheless, it demonstrates how useful transcript data can be for such prediction tasks in contrast to information on students prior to college. We demonstrate that using subsets of data from registrar databases (in this case, aggregates of students’ first year) can be nearly as effective for predictions as hand-generating a wide swath of features from different institutional data sources.

7. ACKNOWLEDGMENTS

The authors would like to thank the data stewards at the University of Washington for their assistance in obtaining the data used in this work.

8. REFERENCES

- [1] E. Aguiar, N. V. Chawla, J. Brockman, G. A. Ambrose, and V. Goodrich. Engagement vs performance: using electronic portfolios to predict first semester engineering student retention. In *Proceedings of the 4th International Conference on Learning Analytics And Knowledge*, pages 103–112. ACM, 2014.
- [2] L. Aulck, R. Aras, L. Li, C. L’Heureux, P. Lu, and J. West. STEM-ming the tide: Predicting STEM attrition using student transcript data. *SIGKDD’s Machine Learning for Education Workshop*, 2017.
- [3] L. Aulck, N. Velagapudi, J. Blumenstock, and J. West. Predicting student dropout in higher education. *ICML’s Machine Learning in Social Good Applications Workshop*, 2016.

- [4] R. S. Baker and P. S. Inventado. Educational data mining and learning analytics. In *Learning Analytics*, pages 61–75. Springer, 2014.
- [5] J. Bayer, H. Bydzovská, J. Géryk, T. Obsivac, and L. Popelinsky. Predicting drop-out from social behaviour of students. In *Proceedings of the 5th International Conference on Educational Data Mining*, 2012.
- [6] A. F. Cabrera, A. Nora, and M. B. Castaneda. The role of finances in the persistence process: A structural model. *Research in Higher Education*, 33(5):571–593, 1992.
- [7] A. F. Cabrera, A. Nora, and M. B. Castaneda. College persistence: Structural equations modeling test of an integrated model of student retention. *The journal of higher education*, 64(2):123–139, 1993.
- [8] A. L. Caison. Analysis of institutionally specific retention research: A comparison between survey and institutional database methods. *Research in Higher Education*, 48(4):435–451, 2007.
- [9] A. P. Carnevale, N. Smith, and J. Strohl. Recovery: Job growth and education requirements through 2020. 2013.
- [10] G. W. Dekker, M. Pechenizkiy, and J. M. Vleeshouwers. Predicting students drop out: A case study. *International Working Group on Educational Data Mining*, 2009.
- [11] D. Delen. Predicting student attrition with data mining methods. *Journal of College Student Retention: Research, Theory & Practice*, 13(1):17–35, 2011.
- [12] C. Demetriou and A. Schmitz-Sciborski. Integration, motivation, strengths and optimism: Retention theories past, present and future. In *Proceedings of the 7th National Symposium on Student Retention, Charleston, SC*, pages 300–312, 2011.
- [13] S. Halawa, D. Greene, and J. Mitchell. Dropout prediction in MOOCs using learner activity features. *Experiences and best practices in and around MOOCs*, 7, 2014.
- [14] D. Hossler. Managing student retention: Is the glass half full, half empty, or simply empty? *College and University*, 81(2):11–14, 2006.
- [15] W. E. Hudson Sr. Can an early alert excessive absenteeism warning system be effective in retaining freshman students? *Journal of College Student Retention: Research, Theory & Practice*, 7(3):217–226, 2005.
- [16] S. M. Jayaprakash, E. W. Moody, E. J. Lauría, J. R. Regan, and J. D. Baron. Early alert of academically at-risk students: An open source analytics initiative. *Journal of Learning Analytics*, 1(1):6–47, 2014.
- [17] N. Johnson. The institutional costs of student attrition. *Delta Cost Project at American Institutes for Research*, 2012.
- [18] Z. J. Kovačić. Early prediction of student success: mining students enrolment data. In *Proceedings of Informing Science & IT Education Conference (InSITE)*, pages 647–665. Citeseer, 2010.
- [19] Y. Luo and Z. A. Pardos. Diagnosing university student subject proficiency and predicting degree completion in vector space. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [20] R. Manrique, B. P. Nunes, O. Marino, M. A. Casanova, and T. Nurmikko-Fuller. An analysis of student representation, representative features and classification algorithms to predict degree dropout. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 401–410. ACM, 2019.
- [21] L. G. Moseley and D. M. Mead. Predicting who will drop out of nursing courses: a machine learning exercise. *Nurse education today*, 28(4):469–475, 2008.
- [22] M. Nagy and R. Molontay. Predicting dropout in higher education based on secondary school performance. In *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, pages 000389–000394. IEEE, 2018.
- [23] D. Niemi and E. Gitin. Using big data to predict student dropouts: Technology affordances for research. In *Proceedings of the International Association for Development of the Information Society (IADIS) International Conference on Cognition and Exploratory Learning in Digital Age*, 2012.
- [24] T. J. Pantages and C. F. Creedon. Studies of college attrition: 1950–1975. *Review of educational research*, 48(1):49–101, 1978.
- [25] S. Ram, Y. Wang, F. Currim, and S. Currim. Using big data for predicting freshmen retention. 2015.
- [26] C. Romero and S. Ventura. Educational data mining: a review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(6):601–618, 2010.
- [27] S. Sadati and N. A. Libre. Development of an early alert system to predict students at risk of failing based on their early course activities. 2017.
- [28] M. Schneider. Finishing the first lap: The cost of first year student attrition in America’s four year colleges and universities. *American Institutes for Research*, 2010.
- [29] J. M. Simons. *A national study of student early alert models at four-year institutions of higher education*. Arkansas State University, 2011.
- [30] W. Spady. Dropouts from higher education: Toward an empirical model. *Interchange*, 2(3):38–62, 1971.
- [31] J. Summerskill. Dropouts from college. In *The American College*. Wiley, New York, 1965.
- [32] D. Thammasiri, D. Delen, P. Meesad, and N. Kasap. A critical assessment of imbalanced class distribution problem: The case of predicting freshmen student attrition. *Expert Systems with Applications*, 41(2):321–330, 2014.
- [33] V. Tinto. Defining dropout: A matter of perspective. *New Directions for Institutional Research*, 1982(36):3–15, 1982.
- [34] V. Tinto. *Leaving college: Rethinking the causes and cures of student attrition*. University of Chicago Press, 1987.
- [35] D. Yang, T. Sinha, D. Adamson, and C. P. Rosé. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Proceedings of the 2013 NIPS Data-driven education workshop*, volume 11, page 14, 2013.

Predictors of Student Satisfaction: A Large-scale Study of Human-Human Online Tutorial Dialogues

Guanliang Chen
Monash University
guanliang.chen@monash.edu

Rafael Ferreira
Universidade Federal Rural de Pernambuco
rafael.ferreira@ed.ac.uk

David Lang
Stanford University
dnlang86@stanford.edu

Dragan Gasevic
Monash University
dragan.gasevic@monash.edu

ABSTRACT

For the development of successful human-agent dialogue-based tutoring systems, it is essential to understand what makes a human-human tutorial dialogue successful. While there has been much research on dialogue-based intelligent tutoring systems, there have been comparatively fewer studies on analyzing large-scale datasets of human-human online tutoring dialogues. A critical indicator of success of a tutoring dialogue can be student satisfaction, which is the focus of the study reported in the paper. Specifically, we used a large-scale dataset, which consisted of over 15,000 tutorial dialogues generated by human tutors and students in a mobile app-based tutoring service. An extensive analysis of the dataset was performed to identify factors relevant to student satisfaction in online tutoring systems. The study also engineered a set of 325 features as input to a Gradient Tree Boosting model to predict tutoring success. Experimental results revealed that (i) in a tutorial dialogue, factors such as efforts spent by both tutors and students, utterance informativeness and tutor responsiveness were positively correlated with student satisfaction; and (ii) Gradient Tree Boosting model could effectively predict tutoring success, especially with utterances from the later period of a dialogue, but more research effort is needed to improve the prediction performance.

Keywords

Intelligent Tutoring Systems, Student Satisfaction, Educational Dialogue Analysis, Gradient Tree Boosting

1. INTRODUCTION

Intelligent tutoring systems (ITS) are computer systems that are designed to act as human tutors and provide personalized instruction or feedback to students in online learning environments [3, 41]. Ultimately, ITS aim at replicating the benefits of one-to-one tutoring in contexts where students cannot receive such tutoring during the learning process

[49]. In the past decades, numerous researchers have been actively involved in the investigation and development of various types of ITS, among which representative examples include AutoTutor [19], BEETLE [16], ASSISTments [25], and Cognitive Tutor [36]. More importantly, these systems have been applied in different educational contexts for hundreds of thousands of students to use and have facilitated student learning. With the aid of ITS, students with an internet connection can receive guidance tailored to their needs and enhance their learning anytime and anywhere. At the same time, instructors and educational institutions can improve their teaching quality and educational programs by analyzing the fine-grained data collected by ITS [1, 3].



Figure 1: A tutoring dialogue example.

A special class of ITS is dialogue-based intelligent tutoring systems such as AutoTutor and BEATTLE, which emphasize the use of human-agent dialogue in one-to-one tutoring. Such systems have been built on advances in psycho-/sociolinguistics, computational linguistics, and natural language processing [14, 35] to create productive learning experiences in human-agent dialogue tutoring.

In line with [47], we argue that the future development of

Guanliang Chen, David Lang, Rafael Ferreira and Dragan Gasevic "Predictors of Student Satisfaction: A Large-scale Study of Human-Human Online Tutorial Dialogues" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 19 - 28

dialogue-based systems can benefit greatly from the analysis of massive datasets collected in online tutoring. Online tutoring has been promoted by a growing number of applications, e.g., Chegg¹, Skooli² and Wyzant³, that offer human-human online tutoring at scale and the learning subjects covered by these applications include math, science, language learning, humanities, etc. Specifically, we are interested in understanding what constitutes successful human-human online tutoring. In this paper, we report on the findings of a study that looked at factors that predict student satisfaction with human-human online tutoring.

The analysis of human-human online tutoring requires consideration of factors that shape the entire tutorial process. Online tutoring, especially helping students to solve problems, is a complex process, in which a student is expected to clearly explain the problems to be solved to a tutor and the tutor is expected to use her knowledge as well as appropriate tutoring strategies to guide the student to solve the problems. The success of such a process depends on many factors, e.g., whether the student clearly explains the problem, whether the tutor asks appropriate questions to guide the student, and whether the tutor provides sufficient emotional support. As shown in the example in Figure 1, the student ended the tutoring session because the tutor did not respond to the student in a timely manner and the student only rated the tutoring session as 1 on the scale of (1,5), which indicated that the student was not satisfied with the tutoring service at all and thus the tutorial dialogue was unsuccessful.

To our knowledge, few studies have attempted to identify the crucial factors that are correlated with the success of a tutoring session. Thus, our work aimed at (i) identifying factors that are correlated with the success of a dialogue-based tutoring session, and further (ii) utilizing the identified features as input to a state-of-the-art machine learning model to predict the tutoring success. Formally, our work was guided by the following research question:

RQ: What factors are related to student satisfaction with online tutoring service?

By investigating the RQ, we expected to (i) help tutors in existing online tutoring systems to better direct their efforts in guiding students, and (ii) inform the design of future dialogue-based ITS.

To this end, we first formulated a set of hypotheses about potential factors that were correlated with the success of a dialogue-based tutoring, which were grounded in previous research findings on online tutoring or relevant educational topics. Then, we conducted an extensive analysis of a large-scale dataset provided by a company offering online tutoring services to students, which contained transcripts of over 15,000 dialogue-based tutoring sessions generated by more than 5,000 students, to test the formulated hypotheses. Based on the analysis, we designed a set of 325 features and used these features as input to a state-of-the-art

machine learning model (i.e., Gradient Tree Boosting) to predict whether a tutorial dialogue would be successful or not.

Experimental results showed that the success of a dialogue-based online tutoring session was associated with factors such as the efforts made by both tutors and students, the informativeness of the utterances, and the sentiment polarity conveyed through the utterances. We further showed that Gradient Tree Boosting was an effective method in predicting the success of tutoring sessions. In particular, we observed that the utterances from the later period of a tutoring session (e.g., the last 20% utterances in a dialogue) could deliver prediction accuracy comparable to that using the whole dialogue as input, and more research effort can be invested to further boost the prediction performance.

2. RELATED WORK

Our work is mainly related to research on educational dialogue analysis [39]. One common theme that has been investigated for years is the development and refinement of a coding scheme for educational dialogue acts [30, 38]. For instance, by building upon the language-as-action theory, [21] proposed a coding scheme that attempts to map utterances to their inherent functions in a dialogue and validated the effectiveness of the proposed scheme in two different learning contexts (one from primary school and the other from secondary school).

Another common strand of work in the field is the investigation of the relationship between tutorial dialogues and student performance [29]. For example, [47] adopted correlation analysis to capture the effects of dialogues on student performance. In particular, the dialogue acts of tutors (e.g., those related to providing explanations) were found to be significantly predictive of students' learning gain. Similarly, [5] found that the choice of corrective tutorial acts adopted by tutors, which serves as an approach to deal with incorrect problem-solving actions, has a significant influence on students' learning gain. In a different vein, [32] measured the quality of a tutorial dialogue with the Classroom Assessment Scoring System-Secondary observational instrument and demonstrated that the quality of educational dialogues was positively associated with student performance. Other relevant works include [13, 17, 24, 31].

Compared to the related works described above, our work distinguished itself in several aspects. Firstly, our work aimed at discovering factors that are related to student satisfaction instead of student performance, though both of them can be regarded as indicative predictors for the success of a tutorial dialogue. Secondly, our work analyzed various types of factors associated with student satisfaction (which are described in Section 3), while prior works have mainly analyzed one or two specific types of factors, e.g., dialogue acts [5, 47] and dialogue quality [32]. Thirdly, the tutorial dialogue dataset used in our work consists of dialogues collected from over 15,000 tutoring sessions initialized by more than 5,000 students, while previous work often used datasets containing a few hundred tutorial dialogues generated by dozens of students.

¹<https://www.chegg.com/>

²<https://www.skooli.com/>

³<https://www.wyzant.com/>

3. APPROACH

In this section, we first describe the dataset we used for analysis. Then, we outline and justify the hypotheses upon which we grounded our work to explore factors that are associated with student satisfaction, and then introduce the method we used to test these hypotheses. Lastly, we describe the machine learning model for predicting student satisfaction.

3.1 Dataset

The dataset used in our work was prepared by an educational technology company that provides on-demand tutoring services via a mobile application and covers topics including mathematics, chemistry, and physics. With the mobile application, a student can take a picture of the problem she encounters or directly write down the problem and select the category to which the problem belongs to. Then, the student will be connected to a professional tutor who can guide the student to solve the problem by leveraging texts and pictures to communicate. Originally, the dataset consisted of dialogues of 18,203 tutoring sessions, which accounted for over 7,000 tutoring hours. To ensure the validity and generalizability of the experimental results, we filtered out dialogues with less than 10 utterances or of duration less than 60 seconds. This was carried out because tutors were unlikely to deliver meaningful tutoring in those sessions.

Table 1: Dataset statistics.

Category	Row ID	Metric	Value
Basic statistics	1	# Sessions	15,756
	2	# Utterances	1,250,270
	3	# Tutors	116
	4	# Students	5,468
	5	Avg. ratings	4.22
Dialogue length	6	Avg. session duration (mins)	28.75
	7	Avg. # utterances / session	79.35
	8	Avg. # words / session	610.87
	9	Avg. # unique words / session	183.80
Activeness	10	Avg. % utterances sent by tutors	57.92
	11	Avg. % words sent by tutors	78.02
	12	Avg. % new words sent by tutors	74.92
Platform experience	13	Avg. # sessions guided by tutors	135.83
	14	Avg. # sessions owned by students	2.88

After filtering, the dataset contained a total of 15,756 dialogues generated by 116 tutors and 5,468 students together, as described in Table 1. It is noteworthy that more than 79% of the dialogues received a rating of 4 or 5 (out of a scale of (1, 5)), as shown in Figure 2, and only about 16% of the dialogues were of rating 1 or 2. This indicates that most of the students were satisfied with the help they received from tutors and those tutoring sessions were successful.

To enable a better understanding of the characteristics of the tutor/student behavior in online tutoring, we further analyzed the dataset from the following perspectives: the length of dialogues, how active tutors/students were in dialogues, and the experiences of tutors/students in using the tutoring platform, and the results are given in Table 1 (Rows 6-14). Firstly, the average duration of all tutorial sessions is about 29 minutes, and we observed that about 50% of the sessions

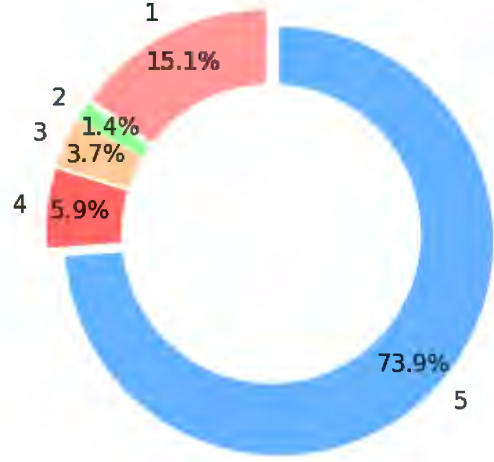


Figure 2: The distribution of student ratings for tutoring sessions.

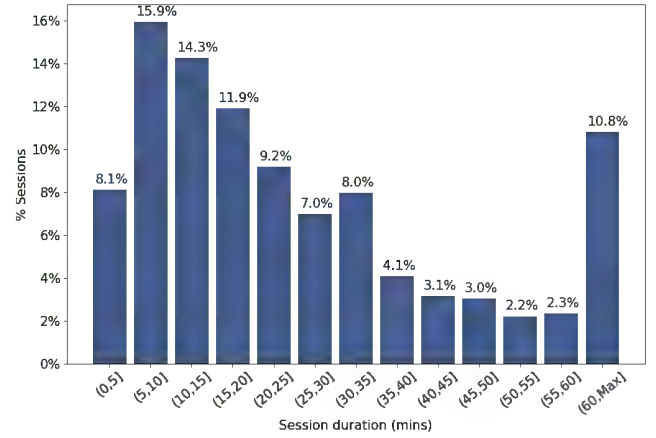


Figure 3: The distribution of the duration of tutoring sessions.

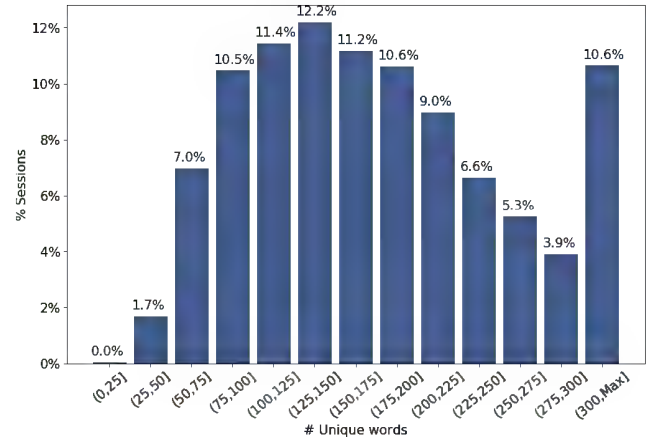


Figure 4: The distribution of the number of unique words in dialogues.

were less than 20 minutes, as shown in Figure 3. On aver-

age, about 80 utterances and 610 words were contained in a dialogue. However, we found only 184 unique words were contained in a dialogue, which was only about 30% of the average number of words used in a dialogue. In fact, most of the dialogues (over 70%) only contained 50~250 unique words (Figure 4). This is in line with previous research finding [6], i.e., people tend to use a relatively small number of words during the conversational process. Also, we observed that tutors were more active than students, i.e., on average, 58% of the utterances in a dialogue were sent by tutors (as shown in Figure 5, tutors sent 50%~70% of the utterances in almost 80% of the dialogues). We had a similar observation when analyzing the average fraction of words sent by tutors, i.e., in over 70% of the dialogues, tutors sent 70%~90% of the words. In particular, 75% of the new words (i.e., words which never appeared in previous utterances) were from tutors. In fact, tutors were in charge of introducing 60%~90% of the new words in 88% of the dialogues (as depicted in Figure 6). This implies that, most of the dialogues were led by the tutors, e.g., tutors were responsible for introducing new concepts to help students solve the problem and guiding students by providing detailed explanations. In terms of platform experience, on average, tutors guided more than 135 sessions, while students only had less than 3 sessions. A detailed analysis revealed that only 43% of the students used the tutoring service for more than once.

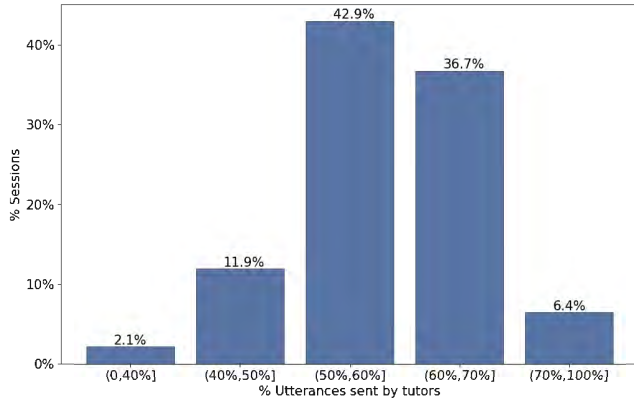


Figure 5: The distribution of the fraction of utterances sent by tutors in dialogues.

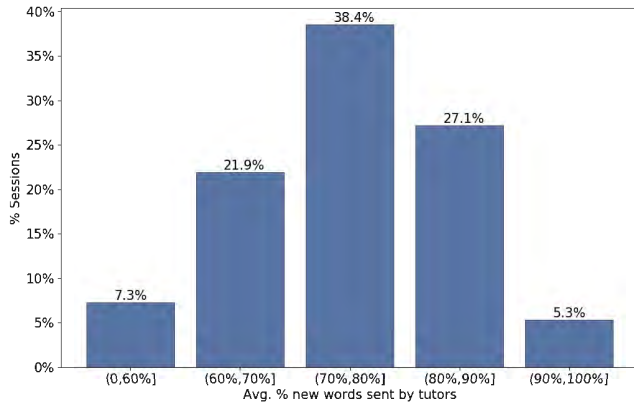


Figure 6: The distribution of the fraction of new words sent by tutors in dialogues.

3.2 Research Hypotheses

Based on prior work, we can make the following hypotheses related to our RQ.

H1 *The more efforts a student/tutor spends in a tutorial dialogue, the more likely the dialogue will be successful.*

The efforts spent by students in learning (e.g., the engagement with course materials) have long been regarded as predictive indicators of their performance [8, 15, 37]. Similarly, we hypothesized that the amount of effort spent by tutors, which directly determines how much help students can receive, also affect students' performance and tutoring success.

H2 *The more informative the utterances sent by a student/tutor are, the more likely a tutorial dialogue will be successful.*

Generally, informative tutoring feedback provided by tutors to students plays a positive role in assisting students in most learning contexts [33]. Here we argue that the informativeness of student utterances also contributes to the success of a tutoring session because it helps tutors quickly understand the difficulties faced by students and correspondingly come up with effective tutoring strategies to help the students.

H3 *The less time a student spends waiting to receive a response from a tutor, the more likely a tutorial dialogue will be successful.*

Previous research on investigating the design and delivery of feedback in online learning environments showed that not only the feedback itself but also the timing of feedback provision impacted student learning [27]. As suggested in [42], effective feedback should be timely so that students can recall the steps of addressing a learning task. Given the fact that a tutoring session is initialized by a student seeking help to solve a problem, this, to a certain extent, implies that the student lacks necessary knowledge but is eager to receive responses from a tutor and solve the problem.

H4 *The higher the lexical entrainment of a tutorial dialogue is, the more likely the dialogue will be successful.*

[6] pointed out that people involved in a conversation tend to coordinate with each other in terms of the words they use (so-called *lexical entrainment*), e.g., both the tutor and the student mentioned the word *triangle* in the dialogue in Figure 1. [34] argued that lexical entrainment is key to facilitate both production and comprehension in dialogues, and more importantly, correlated with task success.

H5 *The less complex the utterances sent by a student/tutor are, the more likely the dialogue will be successful.*

[40] suggested that, in the setting of classroom-based education, tutors should intend to gradually increase the complexity level of their verbal communication with students so as to foster students' level of competency. However, in the setting of online tutoring, in which a tutoring session usually lasts no more than half an hour (as shown in Section 3.1) and the learning task is relatively simple (e.g., solving a math problem), we hypothesized that the complexity level of tutor/student

utterances would be negatively correlated with the tutoring success as complex utterances usually take more time to understand and respond.

- H6** *The more questions a tutor/student asks, the more likely a tutorial dialogue will be successful.*
Previous research demonstrated that questioning is an essential method for tutors to help students build up their understanding and promote effective learning [20, 44, 50]. Likewise, questions asked by a student represent the student's activeness in learning what is unknown to her at the moment and are generally viewed as positively related to her learning performance [43, 45, 46].
- H7** *The more positive sentiment conveyed through the utterances sent by a student/tutor, the more likely a tutorial dialogue will be successful.*
[9, 51] suggested that students' sentiment expressed via forum posts in a MOOC is correlated with the retention rate of the course. This led us to postulate that the success of a tutoring session could be revealed by students' sentiment conveyed in the dialogue utterances. Also, the positive sentiment contained in tutors' utterances, e.g., those used to encourage students, can be indicative of the success of a tutoring session.
- H8** *The more prior tutorial dialogues a student/tutor has, the more likely the current dialogue will be successful.*
On the one hand, the number of prior tutorial dialogues that tutors have can be used to estimate their prior tutoring experience, which is generally believed to have a positive effect on students' learning outcome [48]. On the other hand, if a student has multiple sessions before the current session, this may imply that (i) the student is familiar in using the tutoring platform; and (ii) the tutoring platform has gained the trust of the student by providing satisfactory learning experiences; thus the student repeatedly returns to the platform and uses the service.

3.3 Hypotheses Testing

To test the formulated hypotheses, we first classified tutorial dialogues receiving ratings of 4 or 5 from students as the *Success* group and those of ratings of 1 or 2 as the *Failure* group. Then, we defined a set of metrics to describe the factors investigated in each hypothesis and compared the two groups with Mann-Whitney test on the relevant metrics to test our hypotheses.

For **H1**, we quantified the efforts of tutors/students made in a tutoring session from three perspectives:

- M1 Session duration:** the duration of a tutoring session;
M2 # Utterances: the number of utterances made by a tutor/student;
M3 # Words: the number of words contained in the utterances made by a tutor/student;

To investigate **H2**, we considered four metrics to measure utterance informativeness:

- M4 # Unique words:** the number of unique words contained in utterances sent by a tutor/student;
M5 # Unique concepts: the number of concepts contained in utterances introduced by a tutor/student;
M6 % New words: the fraction of unique words sent by a tutor/student for the first time (so-called *new words*);
M7 % New concepts: the fraction of unique concepts introduced by a tutor/student for the first time (so-called *new concepts*);

Counting the number of unique words (**M4**) was one indicator to measure the informativeness of utterances. Besides, given that both tutors and students often use concepts during the conversational problem-solving process (as shown in Figure 1 where *triangle* was mentioned by both the tutor and the student), and such concepts often bring new information, we also calculated the number of unique concepts (**M5**) to measure the utterance informativeness. As concepts typically appear as nouns, we extracted the nouns contained in an utterance and used them as proxies to capture the mentioned concepts. For this, we use NLTK⁴ to extract nouns from utterances. In addition, we also defined **M6** and **M7** to measure the extent to which the new words/concepts were spoken by tutors/students, as indicators to distinguish the main contributor in bringing new information in a dialogue.

We tested **H3** from two angles:

- M8 Wait time:** the amount of time between a student initialized a request for help and the student was connected to a tutor;
M9 Avg. response time: the average amount of time that a student needed to wait before receiving a reply from a tutor after the student sent an utterance;

To investigate **H4**, we defined the following metric:

- M10 Entrainment:** the score describing the level of entrainment between the tutor utterances and the student utterances;

Inspired by [4], we calculated **M10** as the similarity between the distribution of respective words used by tutors and students. Specifically, we first needed to decide the set of considered words used to calculate entrainment score. [34] suggested that function words (i.e., frequent words like *is*, *do*, *can*) and punctuation marks are important for measuring the degree to which people align with each other in successful dialogues. Therefore, we took all of the words appearing in dialogues into account for calculating **M10** (denoted as *Entrainment (All)*). In addition, as indicated before, both tutors and students often use concepts during the tutoring process and we hypothesized that the entrainment between such concepts was of particular importance to indicate whether a dialogue would be successful or not. Therefore, we also calculated **M10** by only considering concepts (denoted as *Entrainment (Concepts)*). Similar to **H2**, we extracted nouns in utterances and regarded them as the concepts mentioned by tutors and students. With the set of considered words defined, we counted the occurrence of

⁴<https://www.nltk.org>

each word for the utterances made by a tutor/student in a dialogue, respectively, which were represented as two vectors (one for the tutor and the other for the student). Then, we measured the similarity of the two vectors by computing their cosine similarity [22], which served to describe how close the tutor and the student were in terms of the vocabulary they used in a dialogue and thus to indicate to what extent they coordinated the words to each other.

To test **H5-8**, we respectively defined the following metrics:

M11 Complexity: the average complexity of utterances sent by a tutor/student;

M12 # questions: the number of questions asked by a tutor/student;

M13 Sentiment: the overall sentiment polarity scores of utterances sent by a tutor/student;

M14 Platform experience: the number of tutorial dialogues that a tutor/student has prior to the current one;

Specifically, we measured the complexity (**M11**) of an utterance by calculating its Flesch readability score [12], which specify to what extent a piece of text is readable to people by returning a score between [0, 100]. A piece of text with a higher Flesch readability score indicates it is easier to understand. Given that questions often ended with a question mark, we, therefore, computed **M12** as the number of sentences ending with a question mark in the utterances made by a tutor/student. For **M13**, we again use *NLTK* to determine the sentiment polarity score for each utterance. The returned score was of range $(-1, 1)$ with -1 being very negative and 1 being very positive. Then, the values of all utterances sent by a tutor/student were summed up as the overall score of the tutor/student in a dialogue, and the scores of all dialogues in a group were averaged as the final score for the group.

3.4 Tutoring Success Prediction

We aimed to predict whether a tutoring session would be successful or not based on the transcript of the tutorial dialogue, which could be regarded as a binary classification problem. Previous research indicated that there are various techniques that can be used for binary classification problems, such as logistic regression, decision trees, random forests, support vector machines, and neural networks. Gradient Tree Boosting (GTB) [11, 18] is a machine learning technique which can be used for both regression and classification problems. Similar to random forests, GTB is based on the belief that multiple predictors aiming to predict the same target variable will do a better job than any single predictor alone. Therefore, GTB constructs a set of predictors (i.e., decision trees), which are typically trained with a random sub-sample of the data (thus each predictor is slightly different from the others) and the predictions of all predictors are taken into account to give a final prediction. In random forests, the predictors are built independently and the predictions are combined by using techniques like weighted average and majority vote. However, in GTB, the predictors are built sequentially in which the later predictors can learn from mistakes committed by previous predictors

and thus reduce prediction errors. This usually takes less time to reach close to actual predictions. Previous research has demonstrated that GTB is one of the most robust machine learning approaches and can deal with various types of feature data and has reliable predictive power when dealing with unbalanced data (as in our case) [10]. Therefore, we select GTB over other approaches for our prediction task.

We used all of the metrics described in Section 3.3, i.e., M1-14, as features for the Gradient Tree Boosting model. Note that we calculated M2-7, M11-14 for both tutors and students, respectively. M10 was calculated by taking all of the words as well as only the concepts into consideration. In addition, as a common practice in solving text classification problems, we extracted *N-grams* features, i.e., unigrams, bigrams, and trigrams, from the dialogue transcripts. Prior to the N-grams extraction, we preprocessed the dialogue transcripts by removing stopwords (e.g., *can*, *a*, *be*, *is*, *are*), which are of high frequency but seldom carry useful information for classification purposes. To avoid overfitting, we only took the top 100 most frequent unigrams, bigrams, and trigrams into consideration. In total, we designed 325 features.

To set up the experiment, we randomly sampled 80% of the data as the *training* dataset, and the remaining 20% as the *validation* and *testing* datasets (10% for each). To demonstrate the effectiveness of GTB in predicting tutoring success, we selected random forests as the baseline method for comparison. We implemented random forests as well as GTB by using the machine learning library *scikit-learn*⁵ for Python. The parameters for both random forests and GTB were optimized through grid search on the validation dataset, and then we evaluated the models' performance on the testing dataset. In line with previous works on classification problems, especially those dealing with imbalanced data, we adopted three representative metrics for evaluation, i.e., Area Under the Curve (AUC), F1 score, and Cohen's kappa coefficient (Cohen's κ) [23].

In particular, the design of our experiments was guided by the following three questions:

- Q1** How does GTB perform in predicting the success of a tutorial dialogue?
- Q2** How much data is needed to successfully predict tutoring success?
- Q3** Which of the designed features are of particular importance for the prediction performance?

To our knowledge, there have been few works attempting to predict the success of a tutorial dialogue with a large-scale dataset and our work has contributed to this by enabling a better understanding of this problem. By investigating **Q1**, we expected to examine the capability of GTB, which is regarded as a state-of-the-art machine learning technique, in solving this particular prediction task. Previous works demonstrated that the earlier a student is identified as being at risk, the more help a tutor can offer to help the student continue to learn. Similarly, the earlier an unsuccessful tu-

⁵<https://scikit-learn.org/>

toring session can be identified, the more effective intervention can be provided to the student by a tutor. Therefore, by investigating **Q2**, we expected to identify how early an unsuccessful tutoring session can be identified and to shed light on the practicability of using GTB to assist tutors during their interaction with students. Lastly, by answering **Q3**, we expected to examine the contributions made by each type of features for the prediction performance.

4. RESULTS

In this section, we describe the experimental results on hypotheses testing as well as tutoring success prediction.

4.1 Results on Hypotheses Testing

For **H1**, we calculated the mean values of M1-3 for all the dialogues contained in the *Failure* and *Success* groups, which are given in Table 2 (so as the other results for H2-8). Based on the results, we observed that the *Success* dialogues were 50% longer than the *Failure* dialogues (31.20 vs. 19.27). In addition, compared to the *Failure* dialogues, both tutors and students had more utterances in the *Success* dialogues. We had similar observations when comparing the number of words sent by tutors/students in the two groups. Therefore, we conclude that H1 was supported.

Table 2: Results on validating the formulated hypotheses. T represents tutors and S represents students. Significant differences (according to Mann-Whitney test) between *Failure* group and *Success* group are marked with ** ($p < 0.001$).

Hypotheses	Metrics	Failure	Success
H1	Session length (mins) **	19.27	31.20
	# Utterances (T) **	28.79	51.46
	# Utterances (S) **	21.32	35.36
	# Words (T) **	315.95	518.21
	# Words (S) **	82.33	146.44
H2	# Unique words (T) **	117.1	157.12
	# Unique words (S) **	47.43	74.52
	# Unique concepts (T) **	102.84	138.43
	# Unique concepts (S) **	41.73	64.34
	# New words (T) **	76.75	74.38
	# New words (S) **	23.25	25.62
	# New concepts (T) **	76.39	74.66
	# New concepts (S) **	23.61	25.34
H3	Wait time	24.09	24.37
	Avg. response time **	32.93	27.89
H4	Alignment (All) **	0.83	0.86
	Alignment (Concepts) **	0.87	0.89
H5	Complexity (T) **	83.93	85.11
	Complexity (S)	100.71	101.26
H6	# Questions (T) **	10.34	17.14
	# Questions (S) **	1.85	4.05
H7	Sentiment (T) **	4.58	9.38
	Sentiment (S) **	1.54	3.32
H8	Experience (T)	160.66	162.56
	Experience (S) **	9.11	12.67

To validate **H2**, we computed M4-7 over all the utterances sent by a tutor(student) in a dialogue and summed up the values to measure how informative the tutor(student) was in the dialogue. Then, the metric values of all dialogues contained in a group were averaged as the final value. We found that both tutors and students used a higher number of unique words as well as unique concepts (M4-5) in the *Success* group than those in the *Failure* group. In particular, students of *Success* group used about 50% more unique words and concepts than their peers in the *Failure* group. These results suggest that, in order to solve problems, both tutors and students in the *Success* group introduced a greater variety of words during tutoring process and thus were more informative. Interestingly, when inspecting the results of M6-7, we found that the *Success* students introduced a larger fraction of new words as well as new concepts compared to the *Failure* students, and correspondingly the *Success* tutors were less active than the *Failure* tutors in bringing new words and concepts to their dialogues. This motivates us to design further experiments to investigate, during the tutoring process, whether tutors should intentionally encourage students to use more new words and concepts to explain problems as well as their thoughts so as to help students solve the problems. To summarize, the observed results indicate that H2 was supported by the analysis of our dataset.

For **H3**, we only observed a significant difference between the two groups in terms of M9. Compared to *Failure* students, *Success* students spent less time (about 5 seconds) in waiting for responses from tutors. Thus, there was some support for H3.

From the reported results of M10, we concluded that **H4** was supported, i.e., the tutors and students were more likely to align with each other in terms of the words they used in the *Success* group than in the *Failure* group. In particular, when only taking concepts into account, we had a slightly higher entrainment score, which implies a higher degree to which tutors and students coordinated concepts than other words in the tutorial dialogues.

By inspecting the results of M11, we discovered that the utterances made by tutors in the *Success* group were slightly less complex than those in the *Failure* group (85.11 vs. 83.93). However, we did not observe a significant difference between the utterances made by students in the two groups. Therefore, **H5** was only supported for tutors.

For **H6**, the results of M12 were in line with our assumption: both tutors and students asked more questions in successful tutoring sessions than those in unsuccessful ones. Particularly, the *Success* students asked more than two times of questions than *Failure* students. Also, it is important to note that, in both groups, tutors asked many more questions than students (about 4~5 times). This is aligned with our previous findings related to the testing of **H1**: tutors tended to make more efforts than students in tutoring sessions.

For **H7**, we noted that the tutors as well as students in the *Success* group displayed a higher level of positive sentiment than those in the *Failure* group. Also, the tutors were more likely to use words of positive sentiment than students in

both groups. Therefore, we concluded H7 was supported.

Lastly, we observed a significant difference on M14 for students, i.e., the platform experience of the *Success* students was significantly higher than that of the *Failure* students. Thus, we concluded that H8 was only supported for students.

4.2 Results on Tutoring Success Prediction

For Q1 described in 3.4, i.e., whether GTB is capable of predicting the success of a tutoring session, we reported the results of GTB as well as the baseline method (random forests) in Rows 1-2 in Table 3. The results indicated that GTB outperformed random forests on all of three evaluation metrics. This demonstrated the effectiveness of GTB for predicting whether a tutoring session will be successful or not. Particularly, GTB attained an improvement of 21% over random forests in terms of Cohen’s κ , though the value was only 0.4323, which implied that the constructed prediction model achieved a *moderate* performance level [28]. This calls for further research effort in developing more effective prediction models for this particular task.

To answer Q2, we trained GTB by using different portions of the dialogue utterances, i.e., the first 20%/40%/60%/80%. The results are reported in Rows 3-6 in Table 3. To our surprise, even using the first 80% of the data to train GTB, the achieved performance was still much inferior to that of using the whole dataset. For instance, the AUC of using the first 80% data was 0.7368, which was 10% lower than that of using the whole dataset. When it comes to Cohen’s κ , the difference became even larger (28% lower). This may imply that the utterances made by tutors and students in the later stage of a tutoring dialogue (especially the last 20%) possibly contained more information for predicting the success of a tutoring session. This motivated us to train the model with the last 20%/40%/60%/80% of the dialogue utterances and reported their performance in Rows 7-10 in Table 3. The results aligned with our assumption. Specifically, solely using the last 20% data already achieved performance that was comparable to that of using the whole dataset. For AUC, it even achieved slightly better performance. This could be explained by the fact that, at the end of successful tutoring sessions, tutors tended to praise students and acknowledge their achievements and students were likely to express their gratitude to the tutors. As a sanity check, we randomly selected 100 successful and unsuccessful dialogues and checked the last 20% utterances in these dialogues. We found that, most of the successful dialogues contained N-grams like (*appreciate*), (*thanks*), (*well, done*), and (*good, job*), which were seldom observed in unsuccessful dialogues. Undoubtedly, these linguistic features served as good indicators for GTB to determine a dialogue’s success.

Lastly, we conducted an ablation study to answer Q3. An ablation study is a frequently-used method to determine to what extent a feature contributes to the performance of a model. Typically, the contribution of a feature is determined by comparing the performance of a model including the feature with that without the feature. The more the performance decreases after removing a feature, the more contribution the feature makes to the model. Instead of identifying the contributions made by each feature we en-

gineered, we were more interested in determining the contributions made by each type of features, i.e., the eight types of feature investigated in 3.3 (*efforts*, *informativeness*, *responsiveness*, *entrainment*, *complexity*, *questions*, *sentiment* and *platform experience*) and the linguistic features (*unigrams*, *bigrams*, *trigrams*). Therefore, we removed each type of feature at a time and reported the model performance in Row 11-20 in Table 3.

We observed that, the top 3 types of feature that made the most contributions to the prediction performance were *unigrams*, *bigrams*, and *efforts*. This was in line with the observation we had when answering Q2, i.e., the linguistic features were predictive in terms of distinguishing successful dialogues from unsuccessful ones.

5. DISCUSSION AND CONCLUSION

Implications for Online Tutors. Through the extensive analysis presented in Section 3.3, we demonstrated that student satisfaction is correlated with a set of dialogue features, which include (i) the efforts invested by tutors/students; (ii) the informativeness of tutor/student utterances; (iii) the readability level of tutor utterances; (iv) tutor responsiveness; (v) the number of questions asked by tutors/students; (vi) the entrainment level of a tutorial dialogue; (vii) the positive sentiment level of tutor/student utterances; and (viii) students’ experience in using the tutoring service. This may shed some light on how to better direct online tutors’ efforts in guiding students. For example, tutors may consider to provide prompt responses, use more words of positive sentiment and suitable readability level, and ask a suitable number of questions to assist students to solve problems. However, it should be noted that the identified dialogue features (as well as the corresponding tutoring implications) may be correlated with each other, e.g., the increased number of utterances might introduce a higher number of questions asked by tutors/students. Further experiments, e.g., online A/B testing, are needed to verify which factors are actually affecting student satisfaction in this context. Also, it is necessary to further investigate whether there are any other factors contributing to the observed correlation. For example, though we observed that students’ experience (measured by the number of tutoring sessions they had before) is associated with their satisfaction, it is still unclear whether this is because of students’ familiarity in using the platform, which enables them to quickly find a tutor and solve a problem, or because of their established loyalty in using the tutoring service. For the former case, it would be beneficial to develop guidelines to help novice students quickly learn how to use the tutoring service. For the latter case, it would be necessary to scrutinize the tutoring sessions that students had before so as to better investigate the elements contributing to students’ established loyalty for the tutoring platform.

Improvement space for satisfaction prediction. Our study demonstrated that Gradient Tree Boosting model is effective in predicting tutoring success with all of the utterances or the utterances from the later period of a tutorial dialogue as input. However, this might be of little value to improve online tutoring service in the real-world setting, i.e., if an unsuccessful dialogue can only be identified (almost) until the end, there is not much a tutor can do to change

Row ID	Method	Data usage	Features	Compared row	AUC	F1	Cohen's κ
1	Random Forests	All	All	-	0.7913	0.8885	0.3571
2	GTB			1	0.8225 (\uparrow 3.95%)	0.9018 (\uparrow 1.49%)	0.4323 (\uparrow 21.05%)
3	GTB	First 20%	All	2	0.6847 (\downarrow 16.75%)	0.8612 (\downarrow 4.50%)	0.1584 (\downarrow 63.37%)
4		First 40%			0.7088 (\downarrow 13.83%)	0.8705 (\downarrow 3.47%)	0.2098 (\downarrow 51.47%)
5		First 60%			0.7086 (\downarrow 13.84%)	0.8783 (\downarrow 2.60%)	0.2473 (\downarrow 42.79%)
6		First 80%			0.7368 (\downarrow 10.42%)	0.8880 (\downarrow 1.52%)	0.3115 (\downarrow 27.95%)
7	GTB	Last 20%	All	2	0.8275 (\uparrow 0.61%)	0.8735 (\downarrow 3.13%)	0.3901 (\downarrow 9.76%)
8		Last 40%			0.8388 (\uparrow 1.98%)	0.8808 (\downarrow 2.32%)	0.4038 (\downarrow 6.58%)
9		Last 60%			0.8349 (\uparrow 1.51%)	0.8924 (\downarrow 1.04%)	0.4239 (\downarrow 1.93%)
10		Last 80%			0.8271 (\uparrow 0.56%)	0.8882 (\downarrow 1.51%)	0.3754 (\downarrow 13.17%)
11	GTB	All	w/o Efforts	2	0.8217 (\downarrow 0.09%)	0.8887 (\downarrow 1.45%)	0.3749 (\downarrow 13.29%)
12			w/o Infomativity		0.8145 (\downarrow 0.97%)	0.8965 (\downarrow 0.58%)	0.4032 (\downarrow 6.73%)
12			w/o Complexity		0.8205 (\downarrow 0.24%)	0.8961 (\downarrow 0.63%)	0.4018 (\downarrow 7.06%)
13			w/o Responsiveness		0.8170 (\downarrow 0.66%)	0.8974 (\downarrow 0.49%)	0.4180 (\downarrow 3.31%)
14			w/o Questions		0.8196 (\downarrow 0.35%)	0.9002 (\downarrow 0.17%)	0.4213 (\downarrow 2.54%)
15			w/o Entrainment		0.8230 (\uparrow 0.06%)	0.8988 (\downarrow 0.33%)	0.4205 (\downarrow 2.73%)
16			w/o Sentiment		0.8204 (\downarrow 0.26%)	0.8985 (\downarrow 0.36%)	0.4156 (\downarrow 3.86%)
17			w/o Experience		0.8178 (\downarrow 0.57%)	0.8974 (\downarrow 0.49%)	0.4180 (\downarrow 3.31%)
18			w/o Unigrams		0.8045 (\downarrow 2.18%)	0.8818 (\downarrow 2.22%)	0.3692 (\downarrow 14.59%)
19			w/o Bigrams		0.8168 (\downarrow 0.69%)	0.8954 (\downarrow 0.70%)	0.3829 (\downarrow 11.44%)
20			w/o Trigrams		0.8233 (\uparrow 0.10%)	0.8993 (\downarrow 0.27%)	0.4302 (\downarrow 0.49%)

Table 3: Experimental results on tutoring success prediction. The percentage value within brackets indicates the increased/decreased (denoted by \uparrow/\downarrow , respectively) performance of evaluation metrics, which were computed by taking the results of the compared row as a comparison. The results in bold represent the top 3 decreased performance among Rows 10-20.

the situation. Therefore, more research is needed to build effective satisfaction prediction models, especially with only the utterances close to the beginning of a dialogue as input. Since we only engineered relatively shallow linguistic features (i.e., unigrams, bigrams, trigrams) as input for the prediction model, which made much larger contributions to the prediction performance compared to other types of feature, it is worthwhile to explore more in-depth linguistic features (e.g., word/phrase/sentence embedding [2]) to boost the prediction performance. Also, noteworthy is that all the features we designed as input for Gradient Tree Boosting model is derived from dialogue utterances without considering the sequential nature between them. In the future, it would be useful to explore the suitability of time series models to capture the underlying time-aware interaction patterns between tutors and students for this prediction task. In addition, it is recognized that data imbalance (as in our case) can have a big impact on the classification performance [26]. We posit that techniques used to reduce impacts of data imbalance like SMOTE [7] would probably help in future research on this problem.

6. ACKNOWLEDGEMENTS

The authors would like to acknowledge (i) the Institute for Educational Sciences and Grant Number R305B14009, and (ii) the funding of the Faculty of Education at Monash University through the Education Futures initiative.

7. REFERENCES

- [1] A. Alkhatlan and J. Kalita. Intelligent tutoring systems: A comprehensive historical survey with recent developments. *arXiv preprint arXiv:1812.09628*, 2018.
- [2] S. Arora, Y. Liang, and T. Ma. A simple but tough-to-beat baseline for sentence embeddings. In *ICLR 2017*, 2017.
- [3] R. S. Baker. Stupid tutoring systems, intelligent humans. *IJAIED*, 26(2):600–614, 2016.
- [4] L. Benotti, J. Bhaskaran, and D. Lang. Modeling student response times: Towards efficient one-on-one tutoring dialogues. In *NUT@EMNLP*, 2018.
- [5] K. E. Boyer, R. Phillips, M. D. Wallis, M. A. Vouk, and J. C. Lester. Learner characteristics and feedback in tutorial dialogue. In *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, pages 53–61. Association for Computational Linguistics, 2008.
- [6] S. E. Brennan. Lexical entrainment in spontaneous dialog. *Proceedings of ISSD*, 96:41–44, 1996.
- [7] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In T. Theeramunkong, B. Kijsirikul, N. Cercone, and T.-B. Ho, editors, *Advances in Knowledge Discovery and Data Mining*, pages 475–482. Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
- [8] R. M. Carini, G. D. Kuh, and S. P. Klein. Student engagement and student learning: Testing the linkages*. *Research in Higher Education*, 47(1):1–32, Feb 2006.
- [9] D. S. Chaplot, E. Rhim, and J. Kim. Predicting student attrition in moocs using sentiment analysis and neural networks. In *AIED Workshops*, 2015.
- [10] N. V. Chawla. Data mining for imbalanced datasets: An overview. In *Data mining and knowledge discovery handbook*, pages 875–886. Springer, 2009.
- [11] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.
- [12] K. Collins-Thompson. Computational assessment of text readability: A survey of current and future research. *ITL-International Journal of Applied Linguistics*, 165(2):97–135, 2014.
- [13] M. G. Core, J. D. Moore, and C. Zinn. The role of

- initiative in tutorial dialogue. In *EACL*, 2003.
- [14] M. W. Crocker. *Computational psycholinguistics: An interdisciplinary approach to the study of language*, volume 20. Springer Science & Business Media, 2012.
 - [15] J. Davies and M. Graff. Performance in e-learning: online participation and student grades. *British Journal of Educational Technology*, 36(4):657–663, 2005.
 - [16] M. Dzikovska, N. Steinhauser, E. Farrow, J. Moore, and G. Campbell. Beetle ii: Deep natural language understanding and automatic feedback generation for intelligent tutoring in basic electricity and electronics. *IJAIED*, 24(3):284–332, 2014.
 - [17] K. Forbes-Riley, D. Litman, A. Huettner, and A. Ward. Dialogue-learning correlations in spoken dialogue tutoring. In *Proceedings of the 2005 Conference on Artificial Intelligence in Education: Supporting Learning Through Intelligent and Socially Informed Technology*, pages 225–232. Amsterdam, The Netherlands, The Netherlands, 2005. IOS Press.
 - [18] J. H. Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002.
 - [19] A. C. Graesser, K. Wiemer-Hastings, P. Wiemer-Hastings, R. Kreuz, T. R. Group, et al. Autotutor: A simulation of a human tutor. *Cognitive Systems Research*, 1(1):35–51, 1999.
 - [20] J. Hattie. *Visible learning: A synthesis of over 800 meta-analyses relating to achievement*. routledge, 2008.
 - [21] S. Hennessy, S. Rojas-Drummond, R. J. E. Higham, A. M. Marquez, R. M. Rios, R. García-Carrión, O. Torreblanca, and M. J. Barrera. Developing a coding scheme for analysing classroom dialogue across educational contexts. 2016.
 - [22] A. Huang. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008)*, Christchurch, New Zealand, volume 4, pages 9–56, 2008.
 - [23] L. A. Jeni, J. F. Cohn, and F. De La Torre. Facing imbalanced data—recommendations for the use of performance metrics. In *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 245–251. IEEE, 2013.
 - [24] S. Katz, G. O'Donnell, and H. Kay. An Approach to Analyzing the Role and Structure of Reflective Dialogue. *IJAIED*, 11:320–343, 2000. Part I of the Special Issue on Analysing Educational Dialogue Interaction (editor: Rachel Pilkington).
 - [25] K. R. Koedinger, E. A. McLaughlin, and N. T. Heffernan. A quasi-experimental evaluation of an on-line formative assessment and tutoring system. *Journal of Educational Computing Research*, 43(4):489–510, 2010.
 - [26] B. Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, Nov 2016.
 - [27] J. A. Kulik and C.-L. C. Kulik. Timing of feedback and verbal learning. *Review of Educational Research*, 58(1):79–97, 1988.
 - [28] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.
 - [29] N. Maharjan, V. Rus, and D. Gautam. Discovering effective tutorial strategies in human tutorial sessions. In *The Thirty-First International Flairs Conference*, 2018.
 - [30] J. C. Marineau, P. M. Wiemer-Hastings, D. Harter, B. A. Olde, P. Chipman, A. Karnavat, V. Pomeroy, S. Rajan, and A. Graesser. Classification of speech acts in tutorial dialog. 2000.
 - [31] D. E. Meltzer. Relation between students' problem-solving performance and representational format. 2005.
 - [32] H. Muhonen, E. Pakarinen, A.-M. Poikkeus, M.-K. Lerkkanen, and H. Rasku-Puttonen. Quality of educational dialogue and association with students' academic performance. *Learning and Instruction*, 55:67 – 79, 2018.
 - [33] S. Narciss and K. Huth. Fostering achievement and motivation with bug-related tutoring feedback in a computer-based training for written subtraction. *Learning and Instruction*, 16(4):310 – 322, 2006.
 - [34] A. Nenkova, A. Gravano, and J. Hirschberg. High frequency word entrainment in spoken dialogue. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 169–172, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.
 - [35] D. Nguyen, A. S. Doğruöz, C. P. Rosé, and F. de Jong. Computational sociolinguistics: A survey. *Computational linguistics*, 42(3):537–593, 2016.
 - [36] J. F. Pane, B. A. Griffin, D. F. McCaffrey, and R. Karam. Effectiveness of cognitive tutor algebra i at scale. *Educational Evaluation and Policy Analysis*, 36(2):127–144, 2014.
 - [37] T. Phan, S. G. McNeil, and B. R. Robin. Students' patterns of engagement and course performance in a massive open online course. *Computers & Education*, 95:36–44, 2016.
 - [38] R. Pilkington. *Analysing educational discourse: The DISCOUNT scheme*. University of Leeds, Computer Based Learning Unit, 1999.
 - [39] R. Pilkington. Analysing educational dialogue interaction: Towards models that support learning. *IJAIED*, 12:1–7, 2001.
 - [40] S. Podschuweit, S. Bernholt, and M. Brückmann. Classroom learning and achievement: how the complexity of classroom interaction impacts students' learning. *Research in Science & Technological Education*, 34(2):142–163, 2016.
 - [41] J. Psotka, L. D. Massey, and S. A. Mutter. *Intelligent tutoring systems: Lessons learned*. Psychology Press, 1988.
 - [42] J. R. Anderson, A. T. Corbett, K. Koedinger, and R. Pelletier. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4:167–207, 04 1995.
 - [43] L. B. Resnick and L. E. Klopfer. *Toward the Thinking Curriculum: Current Cognitive Research. 1989 ASCD Yearbook*. ERIC, 1989.
 - [44] C. P. Rosé, D. Bhembé, S. Siler, R. K. Srivastava, and K. VanLehn. The role of why questions in effective human tutoring. 2003.
 - [45] A. Taboada and J. T. Guthrie. Growth of cognitive strategies for reading comprehension. *Motivating reading comprehension: Concept-oriented reading instruction*, pages 273–306, 2004.
 - [46] A. Taboada and J. T. Guthrie. Contributions of student questioning and prior knowledge to construction of knowledge from reading information text. *Journal of literacy research*, 38(1):1–35, 2006.
 - [47] A. K. Vail and K. E. Boyer. Identifying effective moves in tutoring: On the refinement of dialogue act annotation schemes. In *Intelligent Tutoring Systems*, 2014.
 - [48] H. J. Van Berkel and D. H. Dolmans. The influence of tutoring competencies on problems, group functioning and student achievement in problem-based learning. *Medical Education*, 40(8):730–736, 2006.
 - [49] K. VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. 2011.
 - [50] J. A. Walsh and B. D. Sattes. *Quality questioning: Research-based practice to engage every learner*. Corwin Press, 2016.
 - [51] M. Wen, D. Yang, and C. P. Rosé. Sentiment analysis in mooc discussion forums: What does it tell us? In *EDM*, 2014.

DAS3H: Modeling Student Learning and Forgetting for Optimally Scheduling Distributed Practice of Skills

Benoît Choffin, Fabrice Popineau, Yolaine Bourda
LRI/CentraleSupélec – University of Paris-Saclay
Gif-sur-Yvette, France
{benoit.choffin, fabrice.popineau, yolaine.bourda}@lri.fr

Jill-Jënn Vie
RIKEN AIP
Tokyo, Japan
vie@jill-jenn.net

ABSTRACT

Spaced repetition is among the most studied learning strategies in the cognitive science literature. It consists in temporally distributing exposure to an information so as to improve long-term memorization. Providing students with an adaptive and personalized distributed practice schedule would benefit more than just a generic scheduler. However, the applicability of such adaptive schedulers seems to be limited to pure memorization, e.g. flashcards or foreign language learning. In this article, we first frame the research problem of optimizing an adaptive and personalized spaced repetition scheduler when memorization concerns the application of underlying multiple skills. To this end, we choose to rely on a student model for inferring knowledge state and memory dynamics on any skill or combination of skills. We argue that no knowledge tracing model takes both memory decay and multiple skill tagging into account for predicting student performance. As a consequence, we propose a new student learning and forgetting model suited to our research problem: DAS3H builds on the additive factor models and includes a representation of the temporal distribution of past practice on the skills involved by an item. In particular, DAS3H allows the learning and forgetting curves to differ from one skill to another. Finally, we provide empirical evidence on three real-world educational datasets that DAS3H outperforms other state-of-the-art EDM models. These results suggest that incorporating both item-skill relationships and forgetting effect improves over student models that consider one or the other.

Keywords

Student modeling, adaptive spacing, memory, knowledge components, q-matrix, optimal scheduling

1. INTRODUCTION

Learners have to manage their studying time wisely: they constantly have to make a trade-off between acquiring new knowledge and reviewing previously encountered learning

material. Considering that learning often involves building on old knowledge (e.g. in mathematics) and that efforts undertaken in studying new concepts may be significant, this issue should not be taken lightly. However, only few school incentive structures encourage long-term retention, making students often favor short-term memorization and poor learning practices [37, 22].

Fortunately, there are simple learning strategies that help students efficiently manage their learning time and improve long-term memory retention at a small cost. Among them, the *spacing* and the *testing* effects have been widely replicated [36, 7] since their discovery in the 19th century. Both of them are recommended by cognitive scientists [24, 46] in order to improve public instruction. The spacing effect states that temporally distributing learning episodes is more beneficial to long-term memory than learning in a single massed study session. The testing effect [35, 5] – also known as *retrieval practice* – basically consists in self-testing after being exposed to new knowledge instead of simply reading the lesson again. This test can take multiple forms: free recall, cued recall, multiple-choice questions, application exercises, and so on. A recent meta-analysis on the testing effect [1] found a strong and positive overall effect size of $g = 0.61$ for testing compared to non-testing reviewing strategies. Another meta-analysis [23] investigated whether learning with retrieval practice could transfer to different contexts and found a medium yet positive overall transfer effect size of $d = 0.40$. Combining both strategies is called *spaced retrieval practice*: temporally distributing tests after a first exposure to knowledge.

Recent research effort has been put on developing adaptive and personalized spacing schedulers for improving long-term retention of flashcards [40, 33, 18]. Compared to non-adaptive schedulers, they show substantial improvement of the learners' retention at immediate and delayed tests [19]. However, and to the best of our knowledge, there is no work on extending these algorithms when knowledge to be remembered concerns the application of underlying skills. Yet, the spacing effect is not limited to vocabulary learning or even pure memorization: it has been successfully applied to the acquisition and generalization of abstract science concepts [44] and to the practice of mathematical skills in a real educational setting [3]. Conversely, most models encountered in knowledge tracing involve multiple skills, but do not model forgetting. The goal of the present article is to start filling this gap by developing a student learning and forgetting

Benoît Choffin, Fabrice Popineau, Yolaine Bourda and Jill-Jënn Vie "DAS3H: Modeling Student Learning and Forgetting for Optimally Scheduling Distributed Practice of Skills" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 29 - 38

model for inferring skills knowledge state and memory dynamics. This model will serve as a basis for the future development of adaptive and personalized skill practice scheduling algorithms for improving learners’ long-term memory.

Our contribution is two-fold. We first frame our research problem for extending the flashcards-based adaptive spacing framework to contexts where memorization concerns the application of underlying skills. In that perspective, students learn and reinforce skill mastery by practicing items involving that skill. We argue that this extension requires new student models to model learning and forgetting processes when multiple skills are involved by a single item. Thus, we also propose a new student model, coined DAS3H, that extends DASH [18, 22] and accounts for memory decay and the benefits of practice when an item can involve multiple knowledge components. Finally, we provide empirical evidence on three publicly available datasets showing that our model outperforms other state-of-the-art student models.

2. RELATED WORK

In this section, we first detail related work on adaptive spacing algorithms before turning to student modeling.

In what follows, we will index students by $s \in \llbracket 1, S \rrbracket$, items (or questions, exercises) by $j \in \llbracket 1, J \rrbracket$, skills or knowledge components (KCs) by $k \in \llbracket 1, K \rrbracket$, and timestamps by $t \in \mathbb{R}^+$ (in days). To be more convenient, we assume that timestamps are encoded as the number of days elapsed since the first interaction with the system. It is sufficient because we only need to know the duration between two interactions. $Y_{s,j,t} \in \{0, 1\}$ gives the binary correctness of student s answering item j at time t . σ is the logistic function: $\forall x \in \mathbb{R}, \sigma(x) = 1/(1 + \exp(-x))$. $KC(\cdot)$ takes as input an item index j and outputs the set of skill indices involved by item j .

Let us quickly detail what we mean by *skill*. In this article, we assimilate skills and knowledge components. Knowledge components are atomistic components of knowledge by which items are tagged. An item may have one or more KCs, and this information is synthesized by a so-called binary q-matrix [41]: $\forall (j, k) \in \llbracket 1, J \rrbracket \times \llbracket 1, K \rrbracket, q_{jk} = \mathbf{1}_{k \in KC(j)}$. We assume that the probability of answering correctly an item j that involves skill k depends on the student’s mastery of skill k ; conversely, we measure skill mastery by the ability of student s to remember skill k and apply it to solve any (possibly unseen) item that involves skill k .

2.1 Adaptive spacing algorithms

Adaptive spacing schedulers leverage the spaced retrieval learning strategy to maximize learning and retention of a set of items. They proceed by sequentially deciding which item to ask the user at any time based on the user’s past study history. Items to memorize are often represented by flashcards, i.e. cards on which one side contains the question (e.g. *When did the Great Fire of London occur?* or *What is the correct translation of “manger” in English?*) and the other side contains the answer.

Early adaptive spacing systems made use of physical flashcards [17] but the advent of computer-assisted instruction made possible the development of electronic flashcards [51],

thus allowing more complex and personalized strategies for optimal reviewing. Nowadays, several adaptive spacing softwares are available to the general public, e.g. Anki¹, SuperMemo², and Mnemosyne³.

Originally, adaptive reviewing systems took decisions based on heuristics and handmade rules [17, 30, 51]. Though maybe effective in practice [20], these early systems lack performance guarantees [40]. Recent research works started to tackle this issue: for instance, Reddy et al. propose a mathematical formalization of the Leitner system and a heuristic approximation used for optimizing the review schedule [32].

A common approach for designing spaced repetition adaptive schedulers consists in modeling human memory statistically and recommending the item whose memory strength is closest to a fixed value θ [22, 18, 20]. Khajah, Lindsey and Mozer found that this simple heuristic is only slightly less efficient than exhaustive policy search in many situations [14]. It has the additional advantage to fit into the notion of “desirable difficulties” coined by Bjork [4]. Pavlik and Anderson [26] use an extended version of ACT-R declarative memory model to build an adaptive scheduler for optimizing item practice (in their case, Japanese-English word pairs) given a limited amount of time. ACT-R is originally capable of predicting item correctness and speed of recall by taking recency and frequency of practice into account. Pavlik and Anderson extend ACT-R to capture the spacing effect as well as item, learner, and item-learner interaction variability. The adaptive scheduler uses the model estimation of memory strength gain at retention test per unit of time to decide when to present each pair of words to a learner.

Other approaches do not rely on any memory model: Reddy, Levine and Dragan formalize this problem as a POMDP (Partially Observable Markov Decision Process) and approximately solve it within a deep reinforcement learning architecture [33]. However, they only test their algorithm on simulated students. A more recent work [40] formalizes the spaced repetition problem with marked temporal point processes and solves a stochastic optimal control problem to optimally schedule spaced review of items. Mettler, Massey and Kellman [19] compare an adaptive spacing scheduler (ARTS) to two fixed spacing conditions. ARTS leverages students’ response times, performance, and number of trials to dynamically compute a priority score for adaptively scheduling item practice. Response time is used as an indicator of retrieval difficulty and thus, learning strength.

Our work can more generally relate to the problem of automatic optimization of teaching sequences. Rafferty et al. formulate this problem as a POMDP planning problem [31]. Whitehill and Movellan build on this work but use a hierarchical control architecture for selecting optimal teaching actions [48]. Lan et al. use contextual bandits to select the best next learning action by using an estimation of the student’s knowledge profile [16]. Many intelligent tutoring systems (ITS) use mastery learning within the Knowledge Tracing [8] framework: making students work on a given

¹<https://apps.ankiweb.net/>

²<https://www.supermemo.com/>

³<https://mnemosyne-proj.org/>

skill until the system infers that they have mastered it.

We can see that the traditional adaptive spacing framework already uses a spaced retrieval practice strategy to optimize the student’s learning time. However, it is not directly adapted to learning and memorization of skills. In this latter case, specific items are the only way to practice one or multiple skills, because we do not have to memorize the content directly. Students who master a skill should be able to generalize to unseen items that also involve that skill. In Section 3, we propose an extension of this original framework in order to apply adaptive spacing algorithms to the memorization of skills.

2.2 Modeling student learning and forgetting

The history of scientific literature on student modeling is particularly rich. In what follows, we focus on the subproblem of modeling student learning and forgetting based on student performance data.

As Vie and Kashima recall [43], two main approaches have been used for modeling student learning and predicting student performance: Knowledge Tracing and Factor Analysis.

Knowledge Tracing [8] models the evolution of a student’s knowledge state over time so as to predict a sequence of answers. The original and still most widespread model of Knowledge Tracing is Bayesian Knowledge Tracing (BKT). It is based on a Hidden Markov Model where the knowledge state of the student is the latent variable and skill mastery is assumed binary. Since its creation, it has been extended to overcome its limits and account for instance for individual differences between students [52]. More recently, Piech et al. replaced the original Hidden Markov Model framework with a Recurrent Neural Network and proposed a new Knowledge Tracing model called Deep Knowledge Tracing (DKT) [29]. Despite a mild controversy concerning the relevance of using deep learning in an educational setting [50], recent works continue to develop this line of research [53, 21].

Contrary to Knowledge Tracing, Factor Analysis does not originally take the order of the observations into account. IRT (Item Response Theory) [42] is the canonical model for Factor Analysis. In its simplest form, IRT reads:

$$\mathbb{P}(Y_{s,j} = 1) = \sigma(\alpha_s - \delta_j)$$

with α_s ability of student s and δ_j difficulty of item j . One of the main assumptions of IRT is that the student ability is static and cannot change over time or with practice. Despite its apparent simplicity, IRT has proven to be a robust and reliable EDM model, even outperforming much more complex architectures such as DKT [49]. IRT can be extended to represent user and item biases with vectors instead of scalars. This model is called MIRT, for Multidimensional Item Response Theory:

$$\mathbb{P}(Y_{s,j} = 1) = \sigma(\langle \alpha_s, \delta_j \rangle + d_j).$$

In this case, α_s and δ_j are d -dimensional vectors, and d_j is a scalar that captures the easiness of item j . $\langle \cdot, \cdot \rangle$ is the usual dot product between two vectors.

More recent works incorporated temporality in Factor Analysis models, by taking practice history into account. For

instance, AFM (Additive Factor Model) [6] models:

$$\mathbb{P}(Y_{s,j} = 1) = \sigma \left(\sum_{k \in KC(j)} \beta_k + \gamma_k a_{s,k} \right)$$

with β_k easiness of skill k and $a_{s,k}$ number of attempts of student s on KC k prior to this attempt. Performance Factor Analysis [27] (PFA) builds on AFM and uses past outcomes of practice instead of simple encounter counts:

$$\mathbb{P}(Y_{s,j} = 1) = \sigma \left(\sum_{k \in KC(j)} \beta_k + \gamma_k c_{s,k} + \rho_k f_{s,k} \right)$$

with $c_{s,k}$ number of correct answers of student s on KC k prior to this attempt and $f_{s,k}$ number of wrong answers of student s on KC k prior to this attempt.

Ekanadham and Karklin take a step further to account for temporality in the IRT model and extend the two-parameter ogive IRT model (2PO model) by modeling the evolution of the student ability as a Wiener process [10]. However, they do not explicitly account for student memory decay.

The recent framework of KTM (Knowledge Tracing Machines) [43] encompasses several EDM models, including IRT, MIRT, AFM, and PFA. KTM’s are based on factorization machines and model the probability of correctness as follows:

$$\mathbb{P}(Y_t = 1) = \sigma \left(\mu + \sum_{i=1}^N w_i x_{t,i} + \sum_{1 \leq i < \ell \leq N} x_{t,i} x_{t,\ell} \langle v_i, v_\ell \rangle \right)$$

where μ is a global bias, N is the number of abstract features, be it item parameters, temporal features, etc., x_t is a sample gathering all features collected at time t : which student answers which item, and information regarding prior attempts, w_i is the bias of feature i and $v_i \in \mathbb{R}^d$ its embedding. The features involved in a sample x_t are typically in sparse number, so this probability can be computed efficiently. In KTM, one can recover several existing EDM models by selecting the appropriate features to consider in the modeling. For instance, if we consider user and item features only, we recover IRT. If we consider the skill features in the q-matrix, and the counter of prior successes and failures at skill level, we recover PFA.

One of the very first works on human memory modeling dates back to 1885 and stems from Ebbinghaus [9]. He models the probability of recall of an item as an exponential function of memory strength and delay since last review. More recently, Settles and Meeder propose an extension of the original exponential forgetting curve model, the half-life regression [38]. They estimate item memory strength as an exponential function of a set of features that contain information on the past practice history and on the item to remember (lexeme tag features, in their case). More sophisticated memory models have also been proposed: for instance ACT-R (Adaptive Character of Thought–Rational) [2] and MCM (Multiscale Context Model) [25].

Walsh et al. [45] offer a comparison of three computational memory models: ACT-R declarative memory model [26],

Predictive Performance Equation (PPE) and a generalization of Search of Associative Memory (SAM). These models differ in how they predict the impact of spacing on subsequent relearning, after a long retention interval. PPE is the only one to predict that spacing may accelerate subsequent relearning (“*spacing accelerated relearning*”) – an effect that was empirically underlined by their experiment. PPE showed also superior fit to experimental data, compared to SAM and ACT-R.

DASH [22, 18] bridges the gap between factor analysis and memory models. DASH stands for Difficulty, Ability, and Student History. Its formulation reads:

$$\mathbb{P}(Y_{s,j,t} = 1) = \sigma(\alpha_s - \delta_j + h_\theta(t_{s,j,1:l}, y_{s,j,1:l-1}))$$

with h_θ a function parameterized by θ (learned by DASH) that summarizes the effect of the $l - 1$ previous attempts where student s reviewed item j ($t_{s,j,1:l-1}$) and the binary outcomes of these attempts ($y_{s,j,1:l-1}$). Their main choice for h_θ is:

$$h_\theta(t_{s,j,1:l}, y_{s,j,1:l-1}) = \sum_{w=0}^{W-1} \theta_{2w+1} \log(1 + c_{s,j,w}) - \theta_{2w+2} \log(1 + a_{s,j,w})$$

with w indexing a set of expanding time windows, $c_{s,j,w}$ is the number of correct outcomes of student s on item j in time window w out of a total of $a_{s,j,w}$ attempts. The time windows w are not disjoint and span increasing time intervals. They allow DASH to account for both learning and forgetting processes. The use of log counts induces diminishing returns of practice inside a given time window and difference of log counts formalizes a power law of practice. The time module h_θ is inspired by ACT-R [2] and MCM [25] memory models.

We can see that Lindsey et al. [18] make use of the additive factor models framework for taking memory decay and the benefits of past practice into account. Their model outperforms IRT and a baseline on their dataset COLT, with an accumulative prediction error metric. To avoid overfitting and making model training easier, they use a hierarchical Bayesian regularization.

To the best of our knowledge, no knowledge tracing model accounts for both multiple skills tagging *and* memory decay. We intend to bridge this gap by extending DASH.

3. FRAMING THE PROBLEM

In our setting, the student learns to master a set of skills by sequentially interacting with an adaptive spacing system. At each iteration, this system selects an item (or exercise, or question) for the student, e.g. *What is $\lim_{x \rightarrow 0} (\sin x)/x$?* This selection is made by optimizing a utility function l that rewards long-term mastery of the set of KCs to learn. Then, the student answers the item and the system uses the correctness of the answer to update its belief concerning the student memory and learning state on the skills involved by the item. Finally, the system provides the student a corrective feedback.

In a nutshell, our present research goal is to maximize mastery and memory of a fixed set of skills among students dur-

ing a given time interval while minimizing the time spent studying.

We rely on the following assumptions:

- information to learn and remember consists in a set of skills⁴ $k \in \llbracket 1, K \rrbracket$;
- skill mastery and memorization of student s at time t is measured by the ability of s to answer an (unseen) item involving that skill, i.e. by their ability to generalize to unseen material;
- students first have access to some theoretical knowledge about skills, but learning happens with retrieval practice;
- items are tagged with one or multiple skills and this information is synthesized inside a binary q-matrix [41];
- students forget: skill mastery decreases as time goes by since last practice of that skill;

Unlike Lindsey et al. [18], we do not assume that items involving skill k are interchangeable: their difficulties, for instance, may differ from one another. Thus, the selection phase is two-fold in that it requires to select the skill to practice *and* the item to present. In theory, there should be at least one item for practicing every skill k ; in practice, one item would be too few, since the student would probably “overfit” on the item. This formalization easily encompasses the flashcards-based adaptive spacing framework: it only requires to associate every item with a distinct skill. This wipes out the need to select an item after the skill.

Different utility functions l can be considered. For instance, Reddy, Levine and Dragan consider both the likelihood of recalling all items and the expected number of items recalled [33]. In our case, the utility function should account for the uncertainty of future items to answer. Indeed, if the goal of the user is to prepare for an exam, the system must take into account that the user will probably have to answer items that they did not train with.

To tackle this problem, like previous work [22, 18], we choose to rely on a student learning and forgetting model. In our case, this model must be able to quantify mastery and memory for any skill or combination of skills. In the next section, we present our main contribution: a new student learning and forgetting model, coined DAS3H.

4. OUR MODEL DAS3H

We now describe our new student learning and forgetting model: DAS3H stands for item Difficulty, student Ability, Skills, and Student Skills practice History, and builds on the DASH model presented in Section 2. Lindsey et al. [18] show that DASH outperforms a hierarchical Bayesian version of IRT on their experimental data, which consist in student-item interactions on a flashcard-based foreign (Spanish) language vocabulary reviewing system. They already talk

⁴These skills may be organized into a graph of prerequisites, but this goes beyond the scope of this article.

about knowledge components, but they use this concept to cluster similar words together (e.g. all conjugations of a verb). Thus, in their setting, an item has exactly one knowledge component; different items can belong to the same knowledge component if they are close enough. As a consequence, their model formulation does not handle multiple skills item tagging, which is common in other disciplines such as in mathematics. Moreover, they assume that the impact of past practice on the probability of correctness does not vary across the skills: indeed, DASH has only two biases per time window w , θ_{2w+1} for past wins and θ_{2w+2} for past attempts. It may be a relevant assumption to prevent overfitting when the number of skills is high, but at the same time it may degrade performance when the set of skills is very diverse and inhomogeneous.

DAS3H extends DASH to items with multiple skills, and allows the influence of past practice on present performance to differ from one skill to another. One could argue that we could aggregate every existing combination of skills into a distinct skill to avoid the burden of handling multiple skills. However, this solution would not be satisfying since the resulting model would for instance not be able to capture item similarities between two items that share all but one skill in common. The use of a representation of multiple skills allows to account for knowledge transfer from one item to another. The item-skill relationships are usually synthesized by a q-matrix and generally require domain experts' labor.

We also leverage the recent Knowledge Tracing Machines framework [43] to enrich the DASH model by embedding the features in d dimensions and model pairwise interactions between those features. So far, KTMs have not been tried with memory features.

In brief, we extend DASH in three ways:

- Extension to handle multiple skills tagging: new temporal module h_θ that also takes the multiple skills into account. The influence of the temporal distribution of past practice and of the outcomes of these previous attempts may differ from one skill to another;
- Estimation of easiness parameters for *each* item j and skill k ;
- Use of KTMs [43] instead of mere logistic regression.

For an embedding dimension of $d = 0$, the quadratic term of KTM is cancelled out and our model DAS3H reads:

$$\mathbb{P}(Y_{s,j,t} = 1) = \sigma(\alpha_s - \delta_j + \sum_{k \in KC(j)} \beta_k + h_\theta(t_{s,j,1:l}, y_{s,j,1:l-1})).$$

Following Lindsey et al. [18], we choose:

$$h_\theta(t_{s,j,1:l}, y_{s,j,1:l-1}) = \sum_{k \in KC(j)} \sum_{w=0}^{W-1} \theta_{k,2w+1} \log(1 + c_{s,k,w}) - \theta_{k,2w+2} \log(1 + a_{s,k,w}).$$

Thus, the probability of correctness of student s on item j at time t depends on their ability α_s , the difficulty of the item δ_j and the sum of the easiness β_k of the skills involved

by item j . It also depends on the temporal distribution and the outcomes of past practice, synthesized by h_θ . In h_θ , w denotes the index of the time window, $c_{s,k,w}$ denotes the amount of times that KC k has been correctly recalled in window w by student s earlier, $a_{s,k,w}$ the amount of times that KC k has been encountered in time window w by student s earlier. Intuitively, h_θ can be seen as a sum of memory strengths, one for each skill involved in item j .

For higher embedding dimensions $d > 0$, in our implementation we use probit as the link function. All features are embedded in d dimensions and their interaction is modeled in a pairwise manner. For a more thorough description of KTMs, see [43]. To implement a model within the KTM framework, one must decide which features to encode in the sparse x vector. In our case, we chose user s , item j , skills $k \in KC(j)$, wins $c_{s,k,w}$ and attempts $a_{s,k,w}$ for each time window w .

Compared to DASH and if we forget about additional parameters induced by the regularization scheme, DAS3H has $(d+1)(K+2W(K-1))$ more feature parameters to estimate. To avoid overfitting, we use additional hierarchical distributional assumptions for the parameters to estimate, as described in the next section.

5. EXPERIMENTS

To evaluate the performance of our model, we compared DAS3H to several state-of-the-art student models on three different educational datasets. These models have been detailed in Section 2.

5.1 Experimental setting

We perform 5-fold cross-validation at the student level for our experiments. This means that the student population is split into 5 disjoint groups and that cross-validation is made on this basis. This evaluation method, also used in [43], has the advantage to show how well an educational data mining model generalizes over previously unseen students.

Following previous work [34, 43] we use hierarchical distributional assumptions when $d > 0$ to help model training and avoid overfitting. More precisely, each feature weight and feature embedding component follows a normal prior distribution $\mathcal{N}(\mu, 1/\lambda)$ where μ and λ follow hyperpriors $\mu \sim \mathcal{N}(0, 1)$ and $\lambda \sim \Gamma(1, 1)$. In their article [18], Lindsey et al. took a similar approach but they assumed that the α_s and the δ_i followed different distributions. Contrary to us, they did not regularize the parameters θ_w associated with the practice history of a student: our situation is different because we have more parameters to estimate than them. We use the same time windows as Lindsey et al. [18]: $\{1/24, 1, 7, 30, +\infty\}$. Time units are expressed in days.

Our models were implemented in Python. Code for replicating our results is freely available on Github⁵. Like Vie and Kashima [43], we used `pywFM`⁶ as wrapper for `libfm`⁷ [34] for models with $d > 0$. We used 300 iterations for the MCMC

⁵<https://github.com/BenoitChoffin/das3h>

⁶<https://github.com/jfloff/pywFM>

⁷<http://libfm.org/>

Gibbs sampler. When $d = 0$, we used the `scikit-learn` [28] implementation of logistic regression with L2 regularization.

We compared DAS3H to DASH, IRT, PFA, and AFM within the KTM framework, for three different embedding dimensions: 0, 5, and 20. When $d > 0$, IRT becomes MIRTb, a variant of MIRT that considers a user bias. We do not compare to DKT, due to the mild controversy over its performance [49, 50]. For DASH, we choose to consider item-specific biases, and not KC-specific biases: in their original setting, Lindsey et al. [18] aggregated items into equivalence classes and trained DASH on this basis. This is not always possible to us because items have in general multiple skill taggings; however, we tested this possibility in Subsection 5.3 but it did not yield better results.

We used three different datasets: ASSISTments 2012-2013 (assist12) [11], Bridge to Algebra 2006-2007 (bridge06) and Algebra I 2005-2006 (algebra05) [39]. The two latter datasets stem from the KDD Cup 2010 EDM Challenge. The main problem for our experiments was that only few datasets that combine both time variables and multiple-KC tagging are publicly available. As a result, only both KDD Cup 2010 datasets have items that involve multiple KCs at the same time. As a further work, we plan to test DAS3H on datasets spanning more diverse knowledge domains and having more fine-grained skill taggings. In ASSISTments 2012-2013, the `problem_id` variable was used for the items and for the KDD Cup datasets, the item variable came from the concatenation of the problem and the step IDs, as recommended by the challenge organizers.

We removed users for whom the number of interactions was less than 10. We also removed interactions with NaN skills, because we feared it would introduce too much noise. For the KDD Cup 2010 datasets, we removed interactions which seemed to be duplicates, i.e. for which the (user, item, timestamp) tuple was duplicated. Finally, we sparsely encoded the features and computed the q-matrices. We detail the dataset characteristics (after preprocessing) in Table 1. The mean skill delay refers to the mean time interval (in days) between two interactions with the same skill, and the mean study period refers to the mean time difference between the last and the first interaction for each student.

5.2 Results

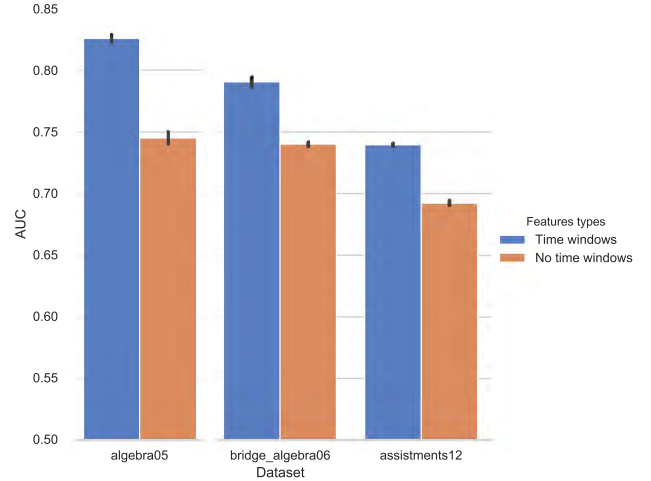
Detailed results can be found in Tables 2, 3 and 4, where mean area under the curve scores (AUC) and mean negative log-likelihood (NLL) are reported for each model and dataset. Accuracy (ACC) is not reported by lack of space. We found that ACC was highly correlated with AUC and NLL; the interested reader can find it on the Github repository containing code for the experiments⁸. Standard deviations over the 5 folds are also reported. We can see that our model DAS3H outperforms all other models on every dataset.

5.3 Discussion

Our experimental results show that DAS3H is able to more accurately model student performance when multiple skill and temporal information is at hand. We hypothesize that

⁸<https://github.com/BenoitChoffin/das3h>

Figure 1: AUC boost when using time windows features instead of regular wins and attempts (all datasets). Higher is better.



this performance gain stems from a more complex temporal modeling of the influence of past practice of skills on current performance.

The impact of the multidimensional embeddings and the pairwise interactions seems to be very small yet unclear, and should be further investigated. An embedding dimension of $d = 20$ is systematically worse or among the worst for DAS3H on every dataset, but with a smaller $d = 5$, the performance is sometimes better than with $d = 0$. An intermediate embedding dimension could be preferable, but our results confirm those of Vie and Kashima [43]: the role of the dimension d seems to be limited.

In order to make more sense of our results, we wanted to know what made DAS3H more predictive than its counterparts. Our hypothesis was that taking the past temporal distribution of practice as well as the outcome of previous encounters with skills allowed the model to capture more complex phenomena than just simple practice, such as forgetting. To test this hypothesis, we performed some ablation tests. We empirically evaluated the difference in terms of AUC on our datasets when time windows features were used instead of regular features for wins and attempts. For each dataset, we compared the mean AUC score of the original DAS3H model with a similar model for which the time windows wins and attempts features were replaced with regular wins and fails counts. Thus, the time module h_θ was replaced with $\sum_{k \in KC(j)} \gamma_k c_{s,k} + \rho_k f_{s,k}$ like in PFA. Since wins, fails and attempts are collinear, it does not matter to replace “wins and attempts” with “wins and fails”. The results are plotted in Figure 1. Mean and standard deviations over 5 folds are reported. We chose an embedding dimension $d = 0$ since it was in general the best on the previous experiments. We observe that using time window features consistently boosts the AUC of the model.

We also wanted to know if assuming that skill practice benefits should differ from one skill to another was a useful assumption. Thus, we compared our original DAS3H formulation to a different version, closer to the DASH formula-

Dataset	Users	Items	Skills	Interactions	Mean correctness	Skills per item	Mean skill delay	Mean study period
assist12	24,750	52,976	265	2,692,889	0.696	1.000	8.54	98.3
bridge06	1,135	129,263	493	1,817,427	0.832	1.013	0.83	149.5
algebra05	569	173,113	112	607,000	0.755	1.363	3.36	109.9

Table 1: Datasets characteristics

model	dim	AUC \uparrow	NLL \downarrow	model	dim	AUC \uparrow	NLL \downarrow
DAS3H	0	0.826 \pm 0.003	0.414 \pm 0.011	DAS3H	5	0.744 \pm 0.002	0.531 \pm 0.001
DAS3H	5	0.818 \pm 0.004	0.421 \pm 0.011	DAS3H	20	0.740 \pm 0.001	0.533 \pm 0.003
DAS3H	20	0.817 \pm 0.005	0.422 \pm 0.007	DAS3H	0	0.739 \pm 0.001	0.534 \pm 0.002
DASH	5	0.775 \pm 0.005	0.458 \pm 0.012	DASH	0	0.703 \pm 0.002	0.557 \pm 0.004
DASH	20	0.774 \pm 0.005	0.456 \pm 0.017	DASH	5	0.703 \pm 0.001	0.557 \pm 0.001
DASH	0	0.773 \pm 0.002	0.454 \pm 0.006	DASH	20	0.703 \pm 0.002	0.557 \pm 0.002
IRT	0	0.771 \pm 0.007	0.456 \pm 0.015	IRT	0	0.702 \pm 0.001	0.558 \pm 0.001
MIRTb	20	0.770 \pm 0.007	0.460 \pm 0.007	MIRTb	20	0.701 \pm 0.001	0.558 \pm 0.001
MIRTb	5	0.770 \pm 0.004	0.459 \pm 0.011	MIRTb	5	0.701 \pm 0.002	0.558 \pm 0.001
PFA	0	0.744 \pm 0.004	0.481 \pm 0.004	PFA	5	0.669 \pm 0.002	0.577 \pm 0.002
AFM	0	0.707 \pm 0.005	0.499 \pm 0.006	PFA	20	0.668 \pm 0.002	0.578 \pm 0.003
PFA	20	0.670 \pm 0.010	1.008 \pm 0.047	PFA	0	0.668 \pm 0.002	0.579 \pm 0.002
PFA	5	0.664 \pm 0.010	1.107 \pm 0.079	AFM	5	0.610 \pm 0.001	0.597 \pm 0.001
AFM	20	0.644 \pm 0.005	0.817 \pm 0.076	AFM	20	0.609 \pm 0.001	0.597 \pm 0.003
AFM	5	0.640 \pm 0.007	0.941 \pm 0.056	AFM	0	0.608 \pm 0.002	0.598 \pm 0.002

Table 2: Performance comparison on the Algebra 2005-2006 (PSLC DataShop) dataset. Metrics are averaged over 5 folds and standard deviations are reported. \uparrow and \downarrow respectively indicate that higher (lower) is better.

Table 3: Performance comparison on the ASSISTments 2012-2013 dataset. Metrics are averaged over 5 folds and standard deviations are reported. \uparrow and \downarrow respectively indicate that higher (lower) is better.

tion, in which all skills share the same parameters θ_{2w+1} and θ_{2w+2} inside a given time window w . We refer to this version of DAS3H as DAS3H_{1p}. The results are given in Table 5. They show that using different parameters for different skills in h_θ increases AUC performance. The AUC gain varies between +0.03 and +0.04. This suggests that some skills have significantly different learning and forgetting curves.

One could argue also that this comparison between DAS3H and DASH is not totally accurate. In their papers, Lindsey et al. cluster similar items together to form disjoint knowledge components. This is not possible to perform directly for both KDD Cup datasets since some items have been tagged with multiple skills. Nevertheless, the ASSISTments 2012-2013 dataset has only single-KC items. To evaluate whether considering the temporal distribution and the outcomes of past practice on the KCs (DASH [KC]) or on the items (DASH [items]) would be better, we compared these two DASH formulations on ASSISTments 2012-2013. Detailed results can be found in Table 6. We see that DASH [items] and DASH [KC] have comparable performance.

Finally, let us illustrate the results of DAS3H by taking two examples of KCs of Algebra I 2005-2006, one for which the estimated forgetting curve slope is steep, the other one for which it is more flat. As a proxy for the forgetting curve slope, we computed the difference of correctness probabilities when a “win” (i.e. a correct outcome when answering an item involving a skill) left a single time window. This difference was computed for every skill, for every couple of time

model	dim	AUC \uparrow	NLL \downarrow
DAS3H	5	0.791 \pm 0.005	0.369 \pm 0.005
DAS3H	0	0.790 \pm 0.004	0.371 \pm 0.004
DAS3H	20	0.776 \pm 0.023	0.387 \pm 0.027
DASH	0	0.749 \pm 0.002	0.393 \pm 0.007
DASH	20	0.747 \pm 0.003	0.399 \pm 0.002
IRT	0	0.747 \pm 0.002	0.393 \pm 0.007
DASH	5	0.747 \pm 0.003	0.399 \pm 0.002
MIRTb	5	0.746 \pm 0.002	0.398 \pm 0.006
MIRTb	20	0.746 \pm 0.004	0.399 \pm 0.007
PFA	20	0.746 \pm 0.003	0.397 \pm 0.004
PFA	5	0.744 \pm 0.007	0.402 \pm 0.007
PFA	0	0.739 \pm 0.003	0.406 \pm 0.008
AFM	5	0.706 \pm 0.002	0.411 \pm 0.004
AFM	20	0.706 \pm 0.002	0.412 \pm 0.004
AFM	0	0.692 \pm 0.002	0.423 \pm 0.006

Table 4: Performance comparison on the Bridge to Algebra 2006-2007 (PSLC DataShop) dataset. Metrics are averaged over 5 folds and standard deviations are reported. \uparrow and \downarrow respectively indicate that higher (lower) is better.

	d	bridge06	algebra05	assist12
DAS3H	0	0.790 \pm 0.004	0.826 \pm 0.003	0.739 \pm 0.001
	5	0.791 \pm 0.005	0.818 \pm 0.004	0.744 \pm 0.002
	20	0.776 \pm 0.023	0.817 \pm 0.005	0.740 \pm 0.001
DAS3H _p	0	0.757 \pm 0.003	0.789 \pm 0.009	0.701 \pm 0.002
	5	0.757 \pm 0.005	0.787 \pm 0.005	0.700 \pm 0.001
	20	0.757 \pm 0.003	0.789 \pm 0.006	0.701 ($<1e-3$)

Table 5: AUC comparison on all datasets between DAS3H and DAS3H_p, a version of DAS3H for which the influence of past practice does not differ from one skill to another. Standard deviations are reported. Higher is better.

DASH	$d = 0$	$d = 5$	$d = 20$
items	0.703 \pm 0.002	0.703 \pm 0.001	0.703 \pm 0.002
KC	0.702 \pm 0.001	0.701 \pm 0.001	0.701 \pm 0.001

Table 6: AUC comparison on ASSISTments 2012-2013 between DASH [items] and DASH [KC]. Standard deviations are reported. Higher is better.

windows, and for every fold. The differences were then averaged over the 5 folds and over the different time windows, yielding for every skill the probability of correctness average decrease when a win leaves a single time window. One of the skills for which memory decays slowly concerns shading an area for which a given value is inferior to a threshold: in average and everything else being equal, the probability of correctness for an item involving this skill decreases by 1.15% when a single “win” leaves a time window. Such a skill is indeed not difficult for a student to master with a few periodic reviews. On the contrary, the skill concerning the application of exponents is more difficult to remember as time goes by: for this KC, the correctness probability decreases by 2.74% when a win leaves a time window. This is more than the double of the previous amount and is consistent with the description of the KC.

In brief, we saw in this section that DAS3H outperforms the other EDM models to which we compared it – including DASH. Using time window features instead of regular skill wins and attempts counts and estimating different parameters for different skills significantly boosts performance. Considering that DAS3H outperforms its ablated counterparts and DASH, these results suggest that including both item-skill relationships and forgetting effect improves over models that consider one or the other. Using multidimensional embeddings, however, did not seem to provide richer feature representations, contrary to our expectations.

Besides its performance, DAS3H has the advantage to be suited to the adaptive skill practice scheduling problem we described in Section 3. Indeed, it encapsulates an estimation of the current mastery of any skill and combination of skills for student s . It can also be used to infer its future evolution and thus, be leveraged to adaptively optimize a personalized skill practice schedule.

6. CONCLUSION AND FUTURE WORK

In this article, we first formulated a research framework for addressing the problem of optimizing human long-term memory of skills. More precisely, the knowledge to be remembered here is *applicative*: we intend to maximize the period during which a human learner will be able to leverage their retention of a skill they practiced to answer an item involving this skill. This framework assumes multiple skills tagging and is adapted to the more common flashcards-based adaptive review schedulers.

We take a student modeling approach to start addressing this issue. As a first step towards an efficient skill practice scheduler for optimizing human long-term memory, we thus propose a new student learning and forgetting model coined DAS3H which extends the DASH model proposed by Lindsey et al. [18]. Contrary to DASH, DAS3H allows each item to depend on an arbitrary number of knowledge components. Moreover, a bias for each skill temporal feature is estimated, whereas DASH assumed that item practice memory decayed at the same rate for every item. Finally, DAS3H is based on the recent Knowledge Tracing Machines model [43] because feature embeddings and pairwise interactions between variables could provide richer models. To the best of our knowledge, KTMs have never been used with memory features so far. Finally, we showed that DAS3H outperforms several state-of-the-art EDM models on three real-world educational datasets that include information on timestamps and KCs. We showed that adding time windows features and assuming different learning and forgetting curves for different skills significantly boosts AUC performance.

This work could be extended in different ways. First, the additive form of our model makes it compensatory. In other terms, if an item j involves two skills k_1 and k_2 , a student could compensate a small practice in k_1 by increasing their practice in k_2 . This is the so-called “explaining away” issue [47]. Using other non-affine models [15] could be relevant. Following Lindsey et al. [18], we used 5 time windows for DAS3H during our experiments: $\{1/24, 1, 7, 30, +\infty\}$. Future work could investigate the impact of alternative sets of time windows – for instance, with more fine-grained time scales. However, one should pay attention not to add too many parameters to estimate.

Future work should also compare DAS3H and DASH to additional student models. For instance, R-PFA [12] (Recent-Performance Factor Analysis) and PFA-decay [13] extend and outperform PFA by leveraging a representation of past practice that puts more weight on more recent interactions. However, they do not explicitly take the temporal distribution of past practice to predict future student performance. Other memory models, such as ACT-R [26] or MCM [25] could also be tested against DAS3H. Latency, or speed of recall, can serve as a proxy of retrieval difficulty and memory strength [19]. It would be interesting to test whether incorporating this information inside DAS3H would result in better model performance.

In a real-world setting, items generally involve multiple skills at the same time. In such a situation, how should one select the next item to recommend a user so as to maximize their long-term memory? The main issue here is that we want to anchor skills in their memory, not specific items. We could think of a two-step recommendation strategy: first, select-

ing the skill k^* whose recall probability is closest to a given threshold (this strategy is consistent with the cognitive psychology literature, as Lindsey et al. recall [18]) and second, selecting an item among the pool of items that involve this skill. However, it could be impossible to find an item that involves *only* this skill k^* . Also, precocious skill reactivations can have a harmful impact on long-term memory [7]. Thus, a strategy could be to compute a score (weighted according to the recall probability of each individual skill) for each skill combination in the q-matrix and to choose the combination for which the score is optimized.

Finally, we tested our model on three real-world educational datasets collected from automatic teaching systems on mathematical knowledge. To experiment with our model, we were indeed constrained in our choice of datasets, since few publicly available of them provide both information on the timestamps and the skills of the interactions. As further work, we intend to test our model on other datasets, from more diverse origins and concerning different knowledge domains. Collecting large, fine-grained and detailed educational datasets concerning diverse disciplines and making them publicly available would more generally allow EDM researchers to test richer models.

Acknowledgements

We would like to warmly thank Pr. Mozer from Univ. of Colorado, Boulder for providing useful details on their papers [22, 18] and allowing us to access the data of their experiment, and to Alice Latimier (LSCP, Paris) for her crucial comments concerning the cognitive science part. This work was funded by Caisse des Dépôts et Consignations, e-Fran program.

7. REFERENCES

- [1] O. O. Adesope, D. A. Trevisan, and N. Sundararajan. Rethinking the use of tests: A meta-analysis of practice testing. *Review of Educational Research*, 87(3):659–701, 2017.
- [2] J. R. Anderson, M. Matessa, and C. Lebiere. ACT-R: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, 12(4):439–462, 1997.
- [3] K. Barzagar Nazari and M. Ebersbach. Distributing mathematical practice of third and seventh graders: Applicability of the spacing effect in the classroom. *Applied Cognitive Psychology*, 33(2):288–298, 2019.
- [4] R. A. Bjork. Memory and metamemory considerations in the training of human beings. *Metacognition: Knowing about knowing*, 185, 1994.
- [5] S. K. Carpenter, H. Pashler, J. T. Wixted, and E. Vul. The effects of tests on learning and forgetting. *Memory & Cognition*, 36(2):438–448, 2008.
- [6] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis—a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pages 164–175. Springer, 2006.
- [7] N. J. Cepeda, E. Vul, D. Rohrer, J. T. Wixted, and H. Pashler. Spacing effects in learning: A temporal ridgeline of optimal retention. *Psychological science*, 19(11):1095–1102, 2008.
- [8] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [9] H. Ebbinghaus. Memory: A contribution to experimental psychology. *Annals of neurosciences*, 20(4):155, 2013.
- [10] C. Ekanadham and Y. Karklin. T-SKIRT: online estimation of student proficiency in an adaptive learning system. In *Machine Learning for Education Workshop at ICML*, 2015.
- [11] M. Feng, N. Heffernan, and K. Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3):243–266, 2009.
- [12] A. Galyardt and I. Goldin. Move your lamp post: Recent data reflects learner knowledge better than older data. *Journal of Educational Data Mining*, 7(2):83–108, 2015.
- [13] Y. Gong, J. E. Beck, and N. T. Heffernan. How to construct more accurate student models: Comparing and optimizing knowledge tracing and performance factor analysis. *International Journal of Artificial Intelligence in Education*, 21(1-2):27–46, 2011.
- [14] M. M. Khajah, R. V. Lindsey, and M. C. Mozer. Maximizing students’ retention via spaced review: Practical guidance from computational models of memory. *Topics in cognitive science*, 6(1):157–169, 2014.
- [15] A. Lan, T. Goldstein, R. Baraniuk, and C. Studer. Dealbreaker: A nonlinear latent variable model for educational data. In *International Conference on Machine Learning*, pages 266–275, 2016.
- [16] A. S. Lan and R. G. Baraniuk. A contextual bandits framework for personalized learning action selection. In *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016*, pages 424–429, 2016.
- [17] S. Leitner. So lernt man lernen [how to learn]. *Freiburg im Breisgau, Germany: Herder*, 1972.
- [18] R. V. Lindsey, J. D. Shroyer, H. Pashler, and M. C. Mozer. Improving students’ long-term knowledge retention through personalized review. *Psychological science*, 25(3):639–647, 2014.
- [19] E. Mettler, C. M. Massey, and P. J. Kellman. A comparison of adaptive and fixed schedules of practice. *Journal of Experimental Psychology: General*, 145(7):897, 2016.
- [20] C. Metzler-Baddeley and R. J. Baddeley. Does adaptive training work? *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*, 23(2):254–266, 2009.
- [21] S. Minn, Y. Yu, M. C. Desmarais, F. Zhu, and J.-J. Vie. Deep knowledge tracing and dynamic student classification for knowledge tracing. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1182–1187. IEEE, 2018.
- [22] M. C. Mozer and R. V. Lindsey. Predicting and improving memory retention: Psychological theory matters in the big data era. In *Big Data in Cognitive Science*, pages 43–73. Psychology Press, 2016.
- [23] S. C. Pan and T. C. Rickard. Transfer of test-enhanced learning: Meta-analytic review and synthesis. *Psychological bulletin*, 144(7):710, 2018.
- [24] H. Pashler, P. M. Bain, B. A. Bottge, A. Graesser, K. Koedinger, M. McDaniel, and J. Metcalfe.

- Organizing instruction and study to improve student learning. IES practice guide. NCER 2007-2004. *National Center for Education Research*, 2007.
- [25] H. Pashler, N. Cepeda, R. V. Lindsey, E. Vul, and M. C. Mozer. Predicting the optimal spacing of study: A multiscale context model of memory. In *Advances in neural information processing systems*, pages 1321–1329, 2009.
- [26] P. I. Pavlik and J. R. Anderson. Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied*, 14(2):101, 2008.
- [27] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis - A new alternative to knowledge tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education, AIED 2009*, pages 531–538, 2009.
- [28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.
- [29] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Advances in neural information processing systems*, pages 505–513, 2015.
- [30] P. Pimsleur. A memory schedule. *The Modern Language Journal*, 51(2):73–75, 1967.
- [31] A. N. Rafferty, E. Brunskill, T. L. Griffiths, and P. Shafto. Faster teaching by POMDP planning. In *International Conference on Artificial Intelligence in Education*, pages 280–287. Springer, 2011.
- [32] S. Reddy, I. Labutov, S. Banerjee, and T. Joachims. Unbounded human learning: Optimal scheduling for spaced repetition. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1815–1824. ACM, 2016.
- [33] S. Reddy, S. Levine, and A. Dragan. Accelerating human learning with deep reinforcement learning. In *NIPS’17 Workshop: Teaching Machines, Robots, and Humans*, 2017.
- [34] S. Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.
- [35] H. L. Roediger III and J. D. Karpicke. Test-enhanced learning: Taking memory tests improves long-term retention. *Psychological science*, 17(3):249–255, 2006.
- [36] H. L. Roediger III and J. D. Karpicke. Intricacies of spaced retrieval: A resolution. In *Successful Remembering and Successful Forgetting*, pages 41–66. Psychology Press, 2011.
- [37] H. L. Roediger III and K. B. McDermott. Remembering what we learn. In *Cerebrum: the Dana Forum on Brain Science*, volume 2018. Dana Foundation, 2018.
- [38] B. Settles and B. Meeder. A trainable spaced repetition model for language learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1848–1858, 2016.
- [39] J. Stamper, A. Niculescu-Mizil, S. Ritter, G. Gordon, and K. Koedinger. Algebra I 2005-2006 and Bridge to Algebra 2006-2007. Development data sets from KDD Cup 2010 Educational Data Mining Challenge. Find them at <http://psl1cdatashop.web.cmu.edu/KDDCup/downloads.jsp>.
- [40] B. Tabibian, U. Upadhyay, A. De, A. Zaregade, B. Schölkopf, and M. Gomez-Rodriguez. Enhancing human learning via spaced repetition optimization. *Proceedings of the National Academy of Sciences*, 116(10):3988–3993, 2019.
- [41] K. K. Tatsuoaka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of educational measurement*, 20(4):345–354, 1983.
- [42] W. J. van der Linden and R. K. Hambleton. *Handbook of modern item response theory*. Springer Science & Business Media, 2013.
- [43] J.-J. Vie and H. Kashima. Knowledge Tracing Machines: Factorization Machines for Knowledge Tracing. In *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, page to appear, 2019.
- [44] H. A. Vlach and C. M. Sandhofer. Distributing learning over time: The spacing effect in children’s acquisition and generalization of science concepts. *Child development*, 83(4):1137–1144, 2012.
- [45] M. M. Walsh, K. A. Gluck, G. Gunzelmann, T. Jastrzembski, M. Krusmark, J. I. Myung, M. A. Pitt, and R. Zhou. Mechanisms underlying the spacing effect in learning: A comparison of three computational models. *Journal of Experimental Psychology: General*, 147(9):1325, 2018.
- [46] Y. Weinstein, C. R. Madan, and M. A. Sumeracki. Teaching the science of learning. *Cognitive Research: Principles and Implications*, 3(1):2, 2018.
- [47] M. P. Wellman and M. Henrion. Explaining “explaining away”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(3):287–292, 1993.
- [48] J. Whitehill and J. Movellan. Approximately optimal teaching of approximately optimal learners. *IEEE Transactions on Learning Technologies*, 11(2):152–164, 2018.
- [49] K. H. Wilson, Y. Karklin, B. Han, and C. Ekanadham. Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. In *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016*, pages 539–544, 2016.
- [50] K. H. Wilson, X. Xiong, M. Khajah, R. V. Lindsey, S. Zhao, Y. Karklin, E. G. Van Inwegen, B. Han, C. Ekanadham, J. E. Beck, et al. Estimating student proficiency: Deep learning is not the panacea. In *Neural Information Processing Systems, Workshop on Machine Learning for Education*, page 3, 2016.
- [51] P. Wozniak and E. J. Gorzelanczyk. Optimization of repetition spacing in the practice of learning. *Acta neurobiologiae experimentalis*, 54:59–59, 1994.
- [52] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International Conference on Artificial Intelligence in Education*, pages 171–180. Springer, 2013.
- [53] J. Zhang, X. Shi, I. King, and D.-Y. Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, pages 765–774. International World Wide Web Conferences Steering Committee, 2017.

Predicting Early and Often: Predictive Student Modeling for Block-Based Programming Environments

Andrew Emerson
North Carolina State University
Raleigh, NC 27695
ajemerso@ncsu.edu

Fernando J. Rodríguez
University of Florida
Gainesville, FL 32611
fjrodriguez@ufl.edu

Bradford Mott
North Carolina State University
Raleigh, NC 27695
bwmott@ncsu.edu

Andy Smith
North Carolina State University
Raleigh, NC 27695
pmsmith4@ncsu.edu

Wookhee Min
North Carolina State University
Raleigh, NC 27695
wmin@ncsu.edu

Kristy Elizabeth Boyer
University of Florida
Gainesville, FL 32611
keboyer@ufl.edu

Cody Smith
North Carolina State University
Raleigh, NC 27695
crsmit16@ncsu.edu

Eric Wiebe
North Carolina State University
Raleigh, NC 27695
wiebe@ncsu.edu

James Lester
North Carolina State University
Raleigh, NC 27695
lester@ncsu.edu

ABSTRACT

Recent years have seen a growing interest in block-based programming environments for computer science education. While these environments hold significant potential for novice programmers, they lack the adaptive support necessary to accommodate students exhibiting a wide range of initial capabilities and dispositions toward computing. A promising approach to addressing this problem is introducing adaptive feedback. This work investigates a key capability for adaptive support: training student models that predict student success in block-based programming activities for novice programmers. The predictive student models utilize four categories of features: prior performance, hint usage, activity progress, and interface interaction. In addition to evaluating the accuracy of these models for multiple block-based programming activities, we also investigate how quickly the models converge to accurate prediction, and we evaluate the additive value of each of the four categories of features. Results show that the predictive models are able to predict whether a student will successfully complete an exercise with high accuracy, as well as converge on this prediction early in the sequence of student interactions.

Keywords

Block-Based Programming, Student Performance Prediction, Predictive Student Models

1. INTRODUCTION

A central thrust of computer science education research in recent years has been improving the recruitment and retention of students into computing-related fields of study [2]. Yet, many undergraduate students face challenges in introductory

programming courses [36], which have been found to be particularly difficult for novice learners [4]. Block-based programming languages are a promising approach to supporting novices because they reduce the need to focus on syntax, thereby reducing cognitive load and encouraging novices to attempt more complex implementations [40]. In contrast to text-based programming languages, students using block-based programming languages have been shown to spend proportionally more time on productive coding [24]. Further, some aspects of block-based code representations, such as the nesting of blocks and the closer alignment with natural language expression, can help students better understand programming concepts [37]. However, despite their benefits, block-based programming environments have typically provided limited support to students, which places a significant burden on students, as well as their instructors and teaching assistants who have limited availability.

Adaptive learning environments have had success in a broad range of subject matters [19, 20, 32]. They also offer a promising vehicle for addressing the complexity of scaffolding and assessing students' programming activities [3, 6, 8, 15]. A key feature of adaptive learning environments is their ability to leverage student models that assess knowledge and skills from observed learning activities and support learning based on the estimated competency levels in real time.

In this paper, we introduce predictive student models that generate a series of predictions of student success on block-based programming activities. As students complete programming activities, predictive student models can predict student performance, thereby enabling adaptive learning environments to make informed decisions on when to proactively provide support and scaffolding to struggling students. This paper presents predictive student models for block-based programming activities that were trained on over 200 undergraduate students' interactions with a block-based programming environment in an introductory engineering course. The models utilized four categories of student programming behavior: *prior performance*, *hint usage*, *activity progress*, and *interface interaction*. The trained models accurately predict student performance on future programming activities. Analyses of the feature sets reveal prior performance to be the most predictive feature early in the

Andrew Emerson, Andy Smith, Cody Smith, Fernando Rodríguez, Wookhee Min, Eric Wiebe, Bradford Mott, Kristy Boyer and James Lester "Predicting Early and Often: Predictive Student Modeling for Block-Based Programming Environments" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 39 - 48

programming exercises, with other features providing more predictive power as the exercises progress.

This paper is structured as follows. Section 2 discusses related work on analyzing student block-based programs and predicting student coding performance. Section 3 describes PRIME, the block-based programming environment that was used to collect the dataset of students’ construction of block-based programs. Section 4 presents the early prediction student modeling framework, as well as an evaluation of the effectiveness of the models, and Section 5 provides a discussion of the results and design implications.

2. RELATED WORK

Hint generation for block-based programming has been the subject of considerable attention. Given the vast solution spaces of programming exercises, data-driven methods of hint generation have been explored extensively. For example, hint generation for Python tutoring [28] as well as comparing the quality of generated hints to expert hints in a block-based programming environment [25] have both shown promise. However, even with high-quality hints, student performance can nevertheless suffer as a result of poor help-seeking behavior, which is prevalent among students who are most in need of assistance [1]. Approaches to hint generation need to address “gaming the system” behaviors, in which students request multiple levels of hints until they receive a “bottom-out” hint [23] providing the answer to the activity. A potential solution is to design a proactive hint generation framework that can monitor students’ progress and deliver proactive support when necessary [5, 12]. The work presented in this paper seeks to enable such proactive feedback by creating predictive models capable of accurately detecting struggling students and doing so as early as possible.

Student modeling in programming environments has largely focused on constructing granular models of student domain knowledge. These approaches seek to apply modeling techniques such as Bayesian Knowledge Tracing [7] to programming exercises, mapping exercises to individual knowledge components to identify which concepts students are struggling with [27], and to enable mastery learning [9, 18]. Related work has sought to leverage large datasets and deep neural architectures to analyze student behaviors and identify struggling students as they complete open-ended programming activities [35]. This work builds on these inner-loop student models by incorporating new features such as prior performance and hint usage to predict student performance.

In addition to work on programming environments, examining student behaviors in open-ended environments has shown promise. For example, Sabourin et al. utilized dynamic Bayesian networks to create early prediction models of student learning in a game-based learning environment [31]. Min et al. used deep-learning techniques and multimodal datasets to recognize student goals in an open-ended game-based learning environment [21, 22]. Others have used textbook annotations to create early prediction models of student learning [38], clustering techniques for early prediction of students interacting in an open-ended exploratory simulation environment [11, 16], and fine-grained analysis of game-based learning behaviors to predict student quitting (or dropout) early [17].

3. METHODS

In this work, we investigate college student interactions with a block-based programming environment using features that capture system-level interactions, prior student data, and programming progress. We first describe the learning environment, the coding activities, and interface design, followed by the study with college students and the coding problem-solving dataset collected for training and analysis.

3.1 PRIME Environment

PRIME is an adaptive learning environment designed to support novices in learning computer science concepts through block-based programming. Students can use PRIME both during class time as well as for lab and homework assignments.

3.1.1 Task Progression Design

The curriculum for PRIME was informed by a review of the syllabi for introductory programming courses from the fifty top-rated undergraduate computer science programs in the US [33]. From this review, we identified the set of topics that are typically covered in the first five units of courses, as well as the order in which they are covered: 1) *Input/Output, Variables, and Loops*, 2) *Functions, Parameters, and Return Values*, 3) *Conditional Execution*, 4) *String Manipulation and Basic Data Structures*, and 5) *Search and Sort Algorithms*. The work presented in this paper focuses on Units 1-3 (Table 1).

Table 1. Computer science curricular coverage.

Unit	Topics
1	PRIME environment tutorial, Input/Output, Numeric data types, Expressions (math), Variables, Iteration (definite)
2	Abstraction, Functions (methods), Parameters, Return Values
3	Boolean data types, Conditionals, Iteration (indefinite), Debugging

Each unit of PRIME is typically covered in a week and consists of multiple sequential activities. Units 1 and 2 consist of seven activities each, with Unit 3 consisting of six activities. Within each unit, activities progressively build upon concepts and require students to build more complex programs to solve increasingly challenging problems.

3.1.2 Interface Design

To support the translation of block-based programs into their text-based equivalents (e.g., Python), PRIME uses Google’s Blockly framework [13] (Figure 1). The primary user interface provides a *Program* panel, a *Console* panel, a *Feedback* panel, and an *Instructions* panel. The *Program* panel consists of a visual coding widget with the block-based coding workspace and toolbox of available blocks.

The default workspace is augmented with a “Start” block, which serves a purpose analogous to the “main” function or method in other programming languages. The toolbox varies for each task, gradually adding more blocks as students complete tasks and are introduced to new topics. This approach is based on prior work indicating that introducing new blocks only as needed may reduce extraneous cognitive

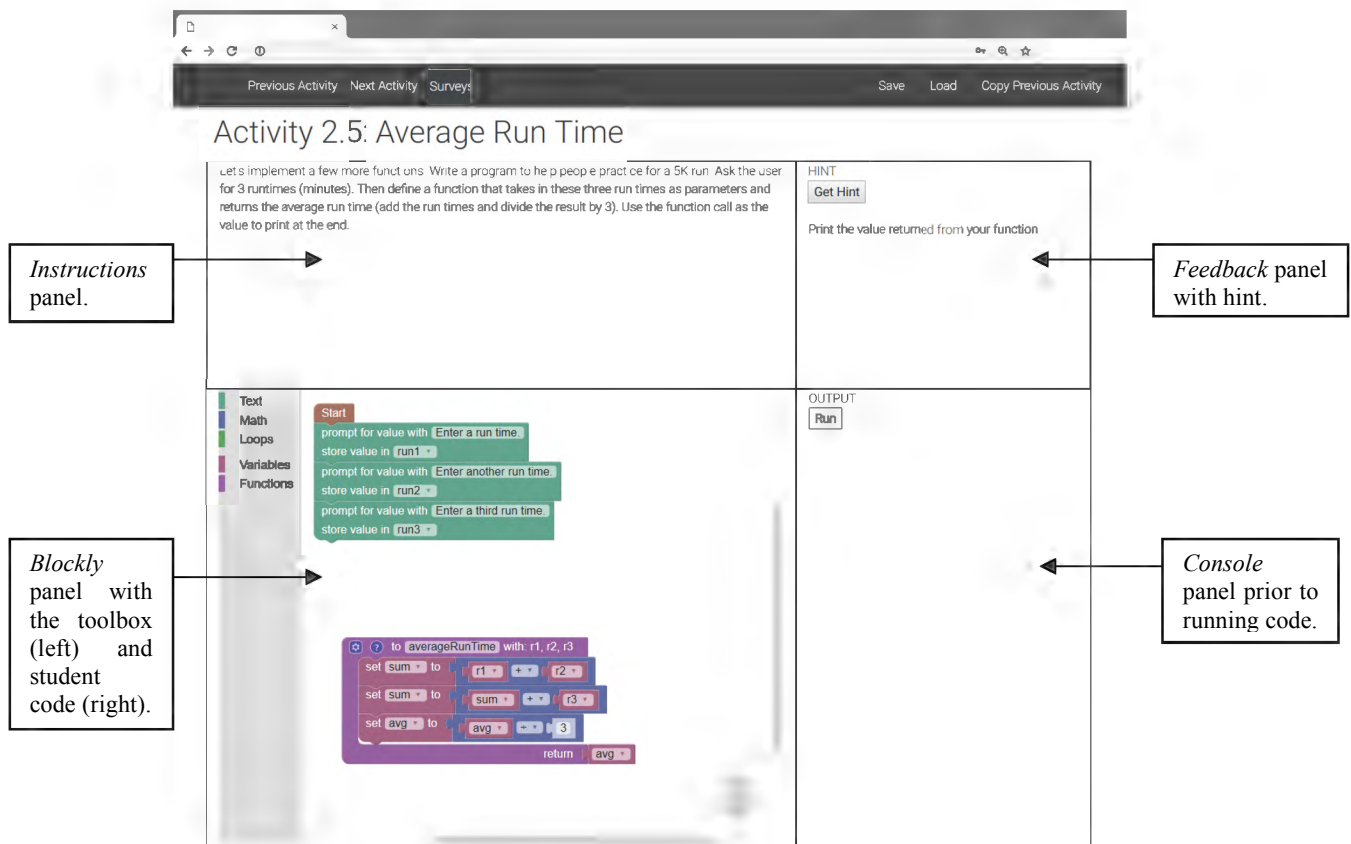


Figure 1. Screenshot of PRIME environment.

load [26] and increase interface usability for novice learners [29].

The *Console* panel contains a “Run” button and shows the output generated from running the program. An input field also appears in this panel if a program prompts the user for input. Finally, the *Instructions* panel contains step-by-step instructions for a given task. This type of instruction format is common for adaptive learning environments [10], though it is rarely found in block-based programming environments. In addition to navigation buttons, this panel also contains positive feedback and links to the next task when a student has successfully completed the current task. Task completeness is checked every time students run their programs and is based primarily on a set of exemplar cases for each activity. Additionally, some activities also check for the presence/absence of certain blocks to ensure an appropriate solution was submitted.

The *Feedback* panel contains the “Get Hint” button, allowing students to request textual hints. Hints are suggestions for minor changes to the program that direct students toward the solution. Hints check various aspects of the student code, including the presence or absence of certain blocks, structural features such as whether code is connected to the “Start” block, and the content of the parameters and fields of certain blocks.

Multiple hints were authored for each activity ($M = 5.40$, $SD = 2.76$) based on common errors identified from prior data collections and pilot testing. The maximum number of hints in an activity is 12, and the minimum authored is 2. An example

of a hint is, “Instead of numbers, you can put other value blocks (like variables) inside the math operation block.” Hints were cast at a sufficiently abstract level that they do not directly provide the solution to an activity but rather nudge students in the right direction. These nudges are designed to assist the student to consider block creations, deletions, or moves that may be advantageous.

Hints are delivered to students in the text panel if they click on the “Get Hint” button in the *Feedback* panel. If no new hints are available, then the button is disabled and cannot be pressed. If there is an available hint different than the one currently displayed, then pressing the button will display that hint. A set of test cases is used to determine which hint is given to the student at a specific point for each activity. The number of test cases passed determines both the specific hint to provide to the student as well as generating the “Next Step” prompt that students receive when completing an intermediate portion of the activity. After requesting a hint, if the student makes changes to satisfy the conditions of the displayed hint, the text of the hint converts to a strikethrough font, visually indicating its status to the student.

3.2 Study Design

Student programming interactions were collected in a study conducted at a large university in the southeastern United States. Participants were students enrolled in two sections of an online introductory course required for all engineering majors. The study sample consisted of 248 students, 222 of which attempted at least one activity.

The average age of the participants was 18, with 31.5% of the group being female. The racial makeup was 75.8% White, 12.9% Asian, 3.2% African American, and 1.2% Hispanic or Latino. The primary major reported was Non-CS Engineering (90.3%), with 6.9% reporting as Computer Science majors and the remaining either Undecided, Math, or Agricultural Science. Of the 222 students, 17.3% reported having prior experience with block-based programming. Of the 222 students who attempted at least one activity, the total number of activities was 2,170 ($M = 9.77$, $SD = 6.24$, Median = 8), and the total number of completed activities was 1,492 ($M = 6.72$, $SD = 4.87$, Median = 4).

3.3 Dataset

The data used in this study was student interaction data collected from students coding with the PRIME block-based programming environment. The data collected for each student consist of actions the student or system took during the course of an attempted activity. For example, the system logs when students perform actions such as requesting a hint, moving a block, creating a block, and other interactions. When a student performs actions relating to a specific block, the system creates an identification number associated with each block to allow easier tracking of specific blocks that the student creates.

As the goal of this work is to predict successful coding activity completion, the activities used for analysis were selected as those with completion rates between 30-70% (i.e., a 70% threshold for the incompleteness/completion rate). Very high completion rates were seen in early activities geared for mastery-oriented introduction to block-based programming. As the activities become more difficult and once the student has been acquainted with the system, the completion rates drop slightly. At the other end of the spectrum, later activities did not have sufficient student attempts for a predictive model to be trained. We therefore focused on the middle activities: 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17, and 18. A summary of the completion rates for each activity is shown in Table 2.

Table 2. Completion Rates for each Activity

Activity	Completed	Attempted	Success Rate (%)
5	57	177	32.2
6	61	126	48.4
7	38	108	35.2
10	59	92	64.1
11	46	81	56.8
12	45	71	63.4
13	36	67	53.7
14	32	60	53.3
15	34	78	43.6
16	38	59	64.4
17	39	56	69.6
18	37	55	67.3

3.4 Feature Families

We formulate the task of predictive student modeling for students' coding activities as a binary classification task. We

define successful completion of a block-based programming activity as a coding activity that the student completes from start to finish and fulfills each of the activity requirements. Completing an activity will only occur once for each attempt of an activity by a student, and it is important to note that there is no time limit for the completion to occur: a student's interaction with an activity can last from the time they start the activity until the semester ends or until he or she has completed the activity. The input for the predictive models is the number of student-activity attempts where each pair denotes a student attempt on a particular activity. There were a total of 1,966 student-activity attempts in the dataset. This count is calculated as the number of student-activity attempts for which the student had at least 20 system-logged actions within the given activity, where an action is a system-logged interaction where the student clicks within the environment, creates/deletes/moves blocks, or interacts with any of the system components, such as the toolbox or "Save Workspace" button. Thus, the inputs of the predictive models are features derived from each of these student-activity attempts. We define the model input vectors formally with four categories of features: *prior performance*, *hint usage*, *activity progress*, and *interface interaction*. The **prior performance** feature is defined as the percentage of activities that a student has completed out of the total activities he or she has attempted up to that point. This feature only considers the activities listed in Table 2. The **hint usage** feature is the total number of times a student has requested a hint for the activity (i.e., pressed the "Get Hint" button).

The **activity progress** features denote the system analysis of the student's code up to a particular point within the activity. We use two features, *test cases passed* and *checkpoints passed*, to represent the student's progress. These features are calculated by evaluating the student's code with expert-designed test cases. The *test cases passed* feature represents the overall test cases required to complete the activity, and the *checkpoints passed* feature consists of finer-grained test cases within the activity. The *checkpoints passed* feature is used to select which hint to give to the student when a hint is requested. In addition, these intermediate-level test cases are used to drive the "Next Step" prompts that the student sees when making incremental progress required to complete the activity.

The final family of features, **interface interaction**, consists of 12 features: *enter button presses*, *last save loads*, *previous exercise code loads*, *next instruction clicks*, *previous instruction clicks*, *code runs*, *workspace saves*, *workspace changes*, *block creations*, *block deletions*, *block moves*, and *user interface clicks in block-display*. These features are logged by the system over the course of a student's attempt at the activity and represent a finer-grained snapshot of the student's problem-solving interactions.

We also use a temporal feature, *time interval*, to introduce a measurement of the student's dynamic progress. We use this to encode sequential interactions and summarize the time interval-based cumulative counts of all other features. The time interval in this work is defined as the 30 second segment of cumulative features up to that point in time (e.g., interval 1 consists of all features within the first 30 seconds, interval 5 consists of the cumulative features up to the first 2 minutes and 30 seconds). There is variation in the maximal interval for

students within the same activity because certain students may take longer to complete (or not complete) the activity. As an example, one student may have spent 2 minutes and 30 seconds on a particular activity, so he or she will have 5 rows of data, each considering the cumulative counts of each described feature. Each row is then indexed by the corresponding time interval, 1 through 5. Before passing the complete input vector into our predictive models, we perform standardization on each feature at the activity level (i.e., subtracting the mean and dividing by the standard deviation).

4. RESULTS

4.1 Activity Completion Predictions

To account for differences in curricular content across activities, we trained a separate model for each activity. We maintained the same set of hyperparameter values for the predictive model for each activity in order to support better generalization to other tasks and to observe patterns spanning all interactions rather than within individual activities. Due to the nature of this predictive task and the fact that the sequential intervals for a particular student-activity attempt are indirectly dependent on one another when comparing the same student, we utilize leave-one-out cross-validation (LOOCV) at the student-level within each activity to validate results found by the predictive models. To report the most accurate results, we averaged the results from the cross-validation process to account for the fact that the student occurring in the test set each iteration does not account for the total distribution of the input data, thereby increasing variance in the results. This validation process was motivated by the occurrence of successive intervals for a given student-activity attempt being related. For example, a student who attempted a specific activity will have cumulative actions in their first interval that will also be counted in the next interval. LOOCV at the student-level thereby prevents data leakage. We adopt logistic regression for interpretability.

Table 3. Classification performance using logistic regression. The results for the majority class baseline (BL), full feature set (Full), and best performing individual family of features (Ind.) are shown.

Activity	Accuracy			F1	
	BL	Full	Ind.	Full	Ind.
5	0.662	0.719	0.666	0.132	0.006
6	0.668	0.751	0.774	0.491	0.469
7	0.708	0.710	0.639	0.281	0.206
10	0.636	0.757	0.815	0.575	0.555
11	0.583	0.762	0.761	0.566	0.563
12	0.747	0.737	0.729	0.680	0.729
13	0.749	0.714	0.664	0.606	0.604
14	0.695	0.704	0.766	0.599	0.611
15	0.607	0.603	0.599	0.417	0.418
16	0.625	0.718	0.694	0.621	0.618
17	0.700	0.865	0.857	0.698	0.683
18	0.757	0.818	0.916	0.701	0.729

We report two metrics: accuracy and F1 score. As a classification problem, it is important to predict both the majority and minority classes at a high rate. In most cases, activity completion is the majority class, but in some cases, the classes are reversed. This occurs primarily when the activities become more difficult, and thus the incompleteness rate is greater than the completion rate for that activity. In reporting these metrics, we show both the results from the full set of features (i.e., all four feature categories), and we choose the best single family of features as a comparison. In addition to these, we compare the results against a baseline of the majority class consisting of every time interval where students spent time on an activity. The label for each interval is the end outcome (completion/incompletion) during their interaction with that activity.

Table 3 summarizes the cross-validation results of the full feature sets and the best performing family of features against the majority class baseline. Across all the activities, prior performance was the best performing family of features in 7 out of 12 activities, or 58% of the time. Of the five remaining activities, interface interaction served as the best performing family once, activity progress twice, and hint usage twice.

4.2 Feature Analysis

After determining the best performing model for the entire set of features, it is informative to determine which of the features held the greatest predictive value. After training the model, we evaluated the feature coefficients of the trained regression to determine the relative importance of each of the features. Both the magnitude and sign of the coefficients can be used for interpretation in this case, as we can determine which features were positive or negative predictors in this classification.

Table 4. Logistic regression model coefficients using the full feature set.

Feature	Mean	SD	Rank
Prior Performance	1.269	0.889	1
Time Interval	-1.176	1.493	2
Test Cases Passed	0.828	0.327	3
Block Deletion	0.648	1.486	4
Block Creation	-0.465	0.692	5
Enter Button Presses	0.458	0.528	6
Checkpoints Passed	-0.277	0.251	7
Save Workspace	-0.232	0.780	8
Next Instruction	0.148	0.663	9
Workspace Change	0.118	0.663	10
User Interface Clicks	0.115	0.989	11
Block Moves	-0.094	0.633	12
Hint Button Presses	0.067	0.799	13
Load Last Save	0.063	0.618	14
Load Previous Exercise Code	0.032	0.112	15
Run Code	-0.013	0.896	16
Previous Instruction	0.004	1.022	17

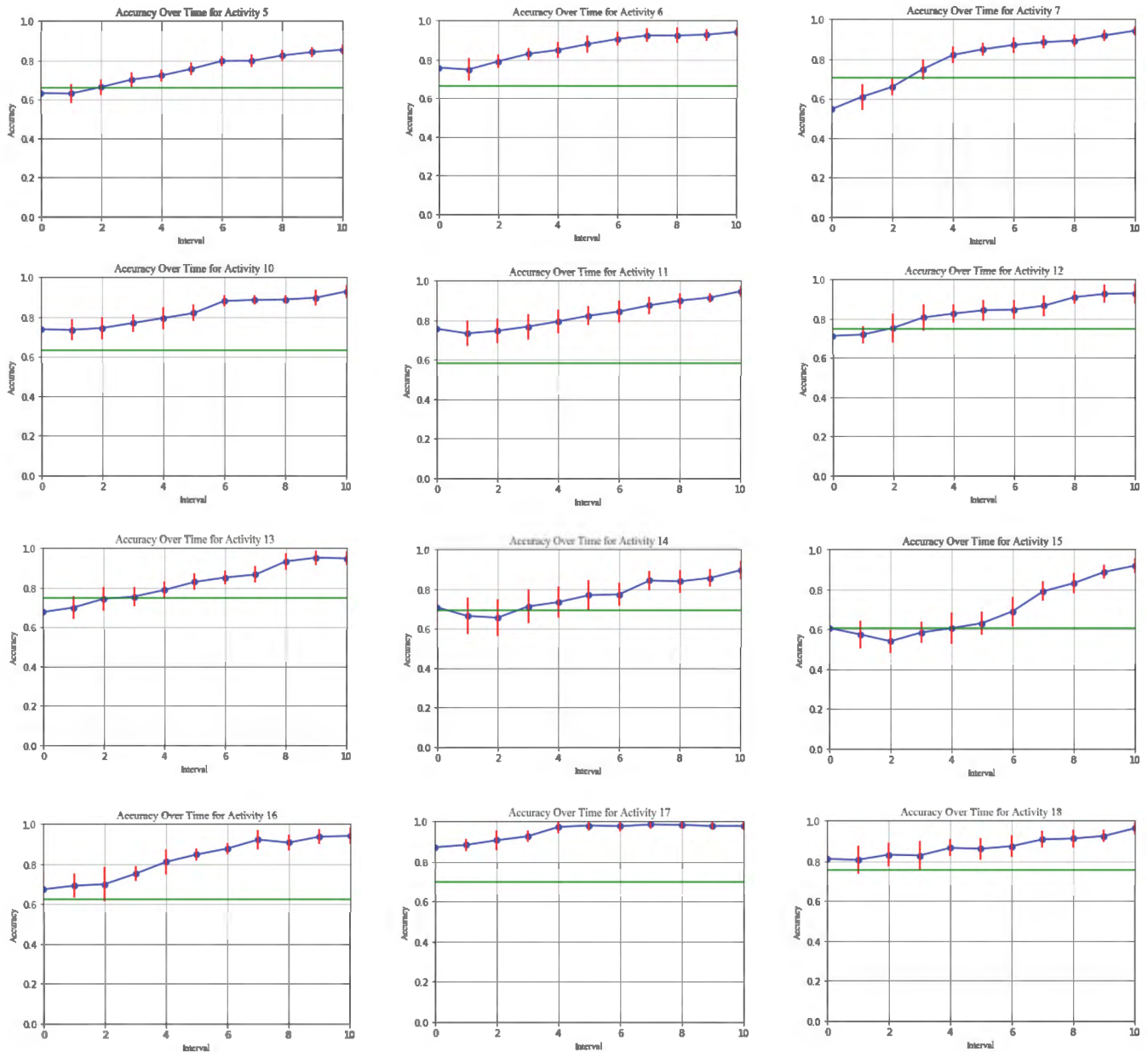


Figure 2. Survival-based analysis of student completion prediction over each interval. The green line represents the baseline for each activity, and the red lines denote the standard deviation of each interval’s average accuracy.

In Table 4, the coefficient results for the full feature set are shown. The four strongest predictors in terms of magnitude were *prior performance*, *time interval*, *test cases passed*, and *block deletion*. Many of the features representing students’ interactions with the block-based environment (e.g., block creations, deletions) provided a strong boost to predictive performance. In addition, features encoding more productivity-based actions, such as *workspace saves* and *checkpoints passed* also provided an improvement to the model. It is worth noting that *time interval* is a strong negative (coefficient direction) predictor, while prior performance was equally as strong of a positive predictor.

4.3 Early Prediction

As a predictive student model observes more student problem-solving actions over time, we would like for its accuracy to improve. A more rapid convergence toward more accurate predictions would mean that an adaptive learning environment could proactively intervene and provide feedback at an earlier stage if the prediction were that the student would not successfully complete an activity. To evaluate this, we performed a survival-based analysis of the predictive models for each interval of each activity. Specifically, we evaluated the performance of our models at each time interval step, where the accuracy at each successive interval includes the students who have already finished interacting with the

activity as correct predictions. The results for this analysis are shown in Figure 2.

For evaluation, we trained the same logistic regression models on each interval for each activity and recorded the number of errors. The accuracy for each interval is then the number of correct predictions divided by the total number of students who attempted that activity (i.e., the total number of samples for the first interval of that activity). Due to the decreasing size of data for each successive interval, we split the data into a 50% train and 50% test set on each interval for each activity, and we took the average performance over 10 randomly generated splits. We then plotted the accuracy over time, noting the standard deviation as the error bars for each interval. We did not perform LOOCV in this analysis because there is at most one interval per student in each activity, so the train/test split will not have data leakage. In other words, splitting the data in half for a train/test split will not have overlapping students in the test set no matter how the split is made.

As noted above, the desired behavior for these predictive models is that accuracy improves over time as the models observe more student problem-solving interaction data. An additional desirable characteristic is that models surpass the baseline at a relatively fast rate. We note that in 8 of the 12 activities, the accuracy of the first interval is at or above that of the baseline (interval-level class majority). For the remaining activities, and those where the accuracy dips below the baseline, the accuracy surpasses the respective baseline at interval 4, which corresponds to 2 minutes of interaction time with PRIME.

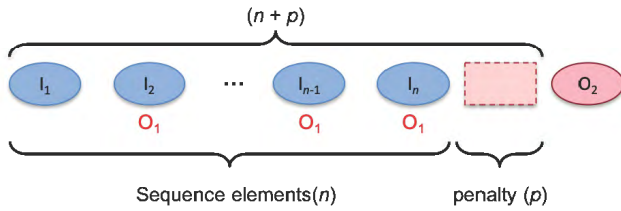


Figure 3. Standardized convergence point metric for sequence prediction analysis.

To quantify the rate at which predictions converged towards an accurate prediction, we also calculated metrics used in the related task of goal recognition for sequence analyses [14, 22]. Specifically, we measured model performance using accuracy rate, convergence rate, convergence point, and standardized convergence point. In this context, *convergence rate* calculates the proportion of sequences where predicted outcome for the final interval is correct. In other words, this metric quantifies how well by the final interval the predictive models can accurately predict whether the student will complete the programming activity. Thus, a higher value for this metric is desirable.

Convergence point refers to the proportion of the sequence of intervals occurring before the predictive model has consistently begun to predict the correct outcome. In other words, this proportion measures a predictive model's ability to converge to an early prediction. This implies that a lower number is more desirable for this metric. The overall convergence point is the average proportion of all sequences of intervals. An issue with using convergence point to measure

how early in a sequence predictions converge is that convergence point is only calculated for sequences where the model successfully predicts the last action in a sequence (i.e., a sequence converged to the correct prediction). *Standardized convergence point* (Figure 3) takes this into account by adding a penalty factor for sequences where the last prediction is incorrect. If the prediction of the outcome (O) for a sequence of intervals (I) does not converge, then its value is calculated as $(n + p)/n$, where p is a penalty factor. For this work we set the penalty factor to 1 because of the relatively short length of the sequences investigated. As with convergence point, a lower value for these metrics is desirable.

In Table 5, we report these metrics for our early prediction models using the same train/test split as mentioned previously. Thus, the sequences of intervals for each student-activity attempt in the test set used as the sequences for these metrics. We average the rates for 10 randomly produced train/test splits to validate the results.

Table 5. Averaged rate results for logistic regression (LR) model.

Metric	
Accuracy Rate	63.96%
Convergence Rate	70.62%
Convergence Point	35.84%
Standardized Convergence Point	57.75%
F1 Score	68.87%

5. DISCUSSION

Four families of features contribute to predictive student modeling. *Prior performance*, *hint usage*, *activity progress*, and *interface interactions* all play an important role in accurately predicting student success in programming activities. The predictive models outperform baselines which use majority class prediction of each individual interval. Using these enhanced models, we found that several features stood out as more predictive.

The *prior performance* feature was the strongest positive (coefficient direction) predictor. If students have successfully completed more of their previous exercises, then they are likely to continue doing so. Because this feature accounts for successes on other activities, when students have just begun attempting activities, there will be no data to inform this feature. This is demonstrated in Table 3 when the F1 score is close to 0 for the individual family of features (Ind.) column. However, when no information is known about a student's prior success (i.e., when they are just starting their interaction), the system can use the other features to account for this. A strong negative predictor of student activity completion was the *time interval*. If a student attempts an activity and begins taking more time, the likelihood of their completing that activity may decrease. This could be due to the student not grasping the underlying concept in which the activity is centered.

Two strong positive predictors were *checkpoints passed* and *test cases passed* (activity progress), denoting how many steps a student has completed in the problem. This is a different measure than time interval, as time interval does not capture

the objective progress the student has made. Therefore, if a model is able to detect how much of the code the student has completed relative to the total code needed for the activity, this could boost predictions. The more incremental checkpoints the system designer uses to assess student code, the more likely this feature will be a strong predictor.

Within the interface interaction family of features, *block creations* served as a negative predictor, while *block deletions* served as a positive predictor. These features are fundamental to understanding a student's code. For systems that do not use built-in test cases, these can serve as core predictive features to use for this prediction task. In PRIME, students can freely create, delete, change, and move blocks according to their believed solution to the activity. Actions such as move and create may reveal a more "trial and error" approach, in which the student is attempting new ideas without knowledge of how these blocks interact. Similarly, actions such as deleting a block and changing a block could indicate when a student has tried a block configuration and no longer believes this to be the correct block configuration. In this case, the student is revising his or her answer, and this could point to a block configuration that is closer to an actual solution.

A surprising result is the fact that *hint button presses* was not one of the strongest predictors. The hint request functionality in this environment guides student problem solving at a conceptual level. Hints are designed to nudge students to consider approaches that may spark a correct move or block creation. Thus, if students request many hints, it may be that they are not getting closer to the correct answer. If they keep requesting hints without successfully completing an activity, this could indicate either a lack of effort or lack of understanding, or perhaps both. For effort, analyses would need to be performed to determine if there is a pattern with other interface interaction features that indicate little attempt on the activity. For understanding, analyses would need to be performed to determine if the student is making little progress, such as is the case in wheel-spinning [34]. One reason that hints did not serve as a stronger predictor could be the fact that there were not a consistent number of hints per activity. In addition, these hints are not hierarchical. In other words, the hints do not utilize a "bottom-out" mechanism that becomes finer-grained as the student requests more hints at the same point in their code. This type of hinting system would allow for students who are clearly experiencing an impasse (or lacking effort) to receive more explicit hints, which would likely change the predictive value of the feature.

5.1 Limitations

In this analysis, predictive models were created to determine if a student will complete a block-based programming activity. We explored the possibility of making this prediction as early as possible. While we quantified this through the improvement of accuracy over time and through the use of convergence rate and convergence point, there are no clear standards or baselines for comparing these results. Without a baseline, it is impossible to fully know how this predictive model performs in relation to other models. When determining the type of model to use, we chose logistic regression due to its relative interpretability. However, other models may have higher performance, especially when tuned appropriately. It will be important to investigate other models in future work.

Another limitation is that this analysis did not fully represent the temporal nature of the data. We created a *time interval* feature to account for 30-second intervals, but we do not treat the actions themselves as sequential features. An alternative to treating the features as sequential could be to create finer-grained intervals (varying time lengths) and to use sequential-based machine learning models, such as probabilistic graphical models or recurrent neural networks. Additionally, though the feature families were chosen to generalize well across learning environments, the underlying features are specific to this environment and may not generalize well. Further investigation is needed to understand the effectiveness of this modelling approach for both other programming environments as well as similarly structured environments from other domains.

A final limitation of this work that should be investigated in future work is level of granularity at which analyses of student code is conducted. One way to analyze student code is to perform static tests, such as in *checkpoints passed* and *test cases passed*. Another method would be to create a new representation of the code and analyze this representation, as the automated code analysis approach presented in [39].

6. CONCLUSION

With increasing interest in block-based programming environments for teaching introductory computer science, programming environments that can provide adaptive support hold considerable promise. In order for these environments to evolve beyond providing on-demand hints, there is a need to develop predictive models that can accurately and quickly identify whether a student will succeed or abandon a given activity.

To explore this potential, we created predictive student models for the PRIME block-based programming environment that were informed by four families of features: prior performance, hint usage, activity progress, and interface interaction. Evaluations showed that the models could predict student activity completion more accurately than baselines, and results also demonstrate that by splitting up student-activity attempt data into time intervals, they can make accurate early predictions. A survival-based analysis showed that by 2 minutes of student interaction time, these models consistently outperform baselines, and prior performance, time interval, and test cases passed were the most predictive features.

In future work, it will be important to investigate modeling frameworks that can better leverage sequential features of the data. Second, it will be important to explore more granular assessment rubrics of student programming artifacts, such as those that might be derived from an evidence-centered design approach [30] to drive the predictive models. Third, exploring models that integrate performance prediction with "sibling" models for help-seeking, off-task-behavior, and wheel-spinning is a promising direction for future work. Here, an ensemble of predictive models could be assembled to most effectively support novice student programming. Finally, it will be important to investigate models that operate in tandem with block-based programming and text-based programming and that best support the transition from block-based to text-based programming as students progress to increasingly complex computational problem-solving tasks.

7. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under Grants DUE-1626235 and DUE-1625908. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

- [1] Aleven, V., Roll, I., McLaren, B.M. and Koedinger, K.R. 2016. Help helps, but only so much: Research on help seeking with intelligent tutoring systems. *International Journal of Artificial Intelligence in Education*. 26, 1, 205–223.
- [2] Beaubouef, T. and Mason, J. 2005. Why the high attrition rate for computer science students: Some thoughts and observations. *ACM SIGCSE Bulletin*. 37, 2, 103–106.
- [3] Blikstein, P. 2011. Using learning analytics to assess students' behavior in open-ended programming tasks. In *Proceedings of the 1st international conference on learning analytics and knowledge*, 110–116.
- [4] Chi, M.T.H. 2005. Commonsense conceptions of emergent processes: Why some misconceptions are robust. *Journal of the Learning Sciences*. 14, 2, 161–199.
- [5] Chi, M.T.H., Siler, S. A., Jeong, H., Yamauchi, T. and Hausmann, R.G. 2001. Learning from human tutoring. *Cognitive Science*. 25, 4, 471–533.
- [6] Corbett, A., Anderson, J.R. and Patterson, E. 1990. Student modeling and tutoring flexibility in the Lisp intelligent tutoring system. In C. Frasson and G. Gauthier (Eds.). *Intelligent tutoring systems: At the crossroads of artificial intelligence and education*. 83–106. Norwood, NJ: Ablex.
- [7] Corbett, A.T. and Anderson, J.R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*. 4, 253–278.
- [8] Corbett, A.T. and Anderson, J.R. 2001. Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement, and attitudes. In *Proceedings of the SIGCHI Conference on Human Computer Interaction*, 245–252.
- [9] Corbett, A.T. and Anderson, J.R. 1992. Student modeling and mastery learning in a computer-based programming tutor. In *Proceedings of the Second International Conference on Intelligent Tutoring Systems*, 413–420.
- [10] Crow, T., Luxton-Reilly, A. and Wuensche, B. 2018. Intelligent tutoring systems for programming education. In *Proceedings of the Twentieth Australasian Computing Education Conference*, 53–62.
- [11] Fratamico, L., Conati, C., Kardan, S. and Roll, I. 2017. Applying a framework for student modeling in exploratory learning environments: Comparing data representation granularity to handle environment complexity. *International Journal of Artificial Intelligence in Education*. 27, 2, 320–352.
- [12] Gerdes, A., Heeren, B., Jeuring, J. and van Binsbergen, L.T. 2016. Ask-Elle: An adaptable programming tutor for Haskell giving automated feedback. *International Journal of Artificial Intelligence in Education*. 27, 1–36.
- [13] Google Blockly - A Visual Programming Editor: 2013.
- [14] Ha, E.Y., Rowe, J.P., Mott, B.W., and Lester, J.C. 2012. Goal recognition with Markov logic networks for player-adaptive games. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2113–2119.
- [15] Ihantola, P., Edwards, S.H., Petersen, A., Sheard, J., Korhonen, A., Spacco, J., Butler, M., Rivers, K., Szabo, C. and Toll, D. 2016. Educational data mining and learning analytics in programming: Literature review and case studies. In *Proceedings of ACM ITiCSE Conference*, 41–63.
- [16] Kardan, S. and Conati, C. 2015. Providing adaptive support in an interactive simulation for learning. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 3671–3680.
- [17] Karumbaiah, S., Baker, R.S. and Shute, V. 2018. Predicting quitting in students playing a learning game. In *Proceedings of the International Conference on Educational Data Mining*, 167–176.
- [18] Kasurinen, J. and Nikula, U. 2009. Estimating programming knowledge with Bayesian knowledge tracing. *ACM SIGCSE Bulletin*. 41, 3, 313.
- [19] Kulik, J.A. and Fletcher, J.D. 2015. Effectiveness of intelligent tutoring systems. *Review of Educational Research*. 37, 1–37.
- [20] Ma, W., Adesope, O., Nesbit, J. and Liu, Q. 2014. Intelligent tutoring systems and learning outcomes: A meta-analysis. *Journal of Educational Psychology*. 106, 4, 901–918.
- [21] Min, W., Frankosky, M.H., Mott, B.W., Rowe, J.P., Wiebe, E., Boyer, K.E. and Lester, J.C. 2015. DeepStealth: Leveraging deep learning models for stealth assessment in game-based learning environments. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence in Education*, 277–286.
- [22] Min, W., Mott, B., Rowe, J., Liu, B. and Lester, J. 2016. Player goal recognition in open-world digital games with long short-term memory networks. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 2590–2596.
- [23] Muldner, K., Burleson, W., Van De Sande, B. and Vanlehn, K. 2011. An analysis of students' gaming behaviors in an intelligent tutoring system: Predictors and impacts. *User Modelling and User-Adapted Interaction*. 21, 1–2, 99–135.
- [24] Price, T.W. and Barnes, T. 2015. Comparing textual and block interfaces in a novice programming environment. In *Proceedings of the Eleventh International Conference on International Computing Education Research*, 91–99.
- [25] Price, T.W., Zhi, R. and Barnes, T. 2017. Hint generation under uncertainty: The effect of hint quality on help-seeking behavior. In *Proceedings of the Eighteenth*

- [26] Renkl, A. and Atkinson, R.K. 2003. Structuring the transition from example study to problem solving in cognitive skill acquisition: A cognitive load perspective. *Educational Psychologist*. 38, 1, 15–22.
- [27] Rivers, K., Harpstead, E. and Koedinger, K. 2016. Learning curve analysis for programming: Which concepts do students struggle with? In *Proceedings of the Twelfth International Computing Education Research Conference*, 143–151.
- [28] Rivers, K. and Koedinger, K.R. 2017. Data-driven hint generation in vast solution spaces: A self-improving Python programming tutor. *International Journal of Artificial Intelligence in Education*. 27, 1, 37–64.
- [29] Rodríguez, F.J., Price, K.M., Isaac, J., Boyer, K.E. and Gardner-McCune, C. 2017. How block categories affect learner satisfaction with a block-based programming interface. In *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2017*, 205.
- [30] Rupp, A., Levy, R., Dicerbo, K.E., Sweet, S.J., Crawford, A. V., Calico, T., Benson, M., Fay, D., Kunze, K.L., Misleve, R.J. and Behrens, J. 2012. Putting ECD into practice: The interplay of theory and data in evidence models within a digital learning environment. *Journal of Educational Data Mining*. 4, 1, 49–110.
- [31] Sabourin, J., Mott, B. and Lester, J. 2013. Utilizing dynamic Bayes nets to improve early prediction models of self-regulated learning. In *Proceedings of the Twenty-First International Conference on User Modeling, Adaptation and Personalization*, 228–241.
- [32] Steenbergen-Hu, S. and Cooper, H. 2013. A meta-analysis of the effectiveness of intelligent tutoring systems on K–12 students’ mathematical learning. *Journal of Educational Psychology*. 105, 4, 970–987.
- [33] The 50 best computer-science and engineering schools in America: 2015. <http://www.businessinsider.com/best-computer-science-engineering-schools-in-america-2015-7/>.
- [34] Wan, H. and Beck, J.E. 2015. Considering the influence of prerequisite performance on wheel spinning. In *Proceedings of the Eighth International Conference on Educational Data Mining*, 129–135.
- [35] Wang, L., Sy, A., Liu, L. and Piech, C. 2017. Learning to represent student knowledge on programming exercises using deep learning. In *Proceedings of the Tenth International Conference on Educational Data Mining*, 324–329.
- [36] Watson, C. and Li, F.W. 2014. Failure rates in introductory programming revisited. In *Proceedings of the Nineteenth Conference on Innovation & Technology in Computer Science*, 39–44.
- [37] Weintrop, D. and Wilensky, U. 2015. Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, 101–110.
- [38] Winchell, A., Mozer, M., Lan, A., Grimaldi, P. and Pashler, H. 2018. Can textbook annotations serve as an early predictor of student learning? In *Proceedings of the Eleventh International Conference on Educational Data Mining*, 431–437.
- [39] Wu, M., Mosse, M., Goodman, N. and Piech, C. 2019. Zero Shot Learning for Code Education : Rubric Sampling with Deep Learning Inference. In *Proceedings of the Thirty-Third International Conference of the Association for Advancement of Artificial Intelligence*.
- [40] Xie, B. and Abelson, H. 2016. Skill progression in MIT app inventor. In *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* 213–217.

Modeling and Experimental Design for MOOC Dropout Prediction: A Replication Perspective

Josh Gardner
Paul G. Allen School of
Computer Science &
Engineering
University of Washington
jpgard@cs.washington.edu

Ryan S. Baker
Graduate School of Education
The University of Pennsylvania
rybaker@upenn.edu

Yuming Yang
Department of Statistics
The University of Michigan
yangym@umich.edu

Christopher Brooks
School of Information
The University of Michigan
brooks@umich.edu

ABSTRACT

Replication of machine learning experiments can be a useful tool to evaluate how both *modeling* and *experimental design* contribute to experimental results; however, existing replication efforts focus almost entirely on modeling alone. In this work, we conduct a three-part replication case study of a state-of-the-art LSTM dropout prediction model. In our first experiment, we replicate the original authors' methodology as precisely as possible in collaboration with the original authors. In a second experiment, we demonstrate that this initial experiment likely overestimates the generalization performance of the proposed model due to the design of its validation. In a third experiment, we attempt to achieve the previously-reported performance in a more difficult, but more relevant, hold-out set design by exploring a large space of model regularization configurations. We demonstrate that we can reduce overfitting and improve generalization performance of the model, but cannot achieve the previously-reported level of performance. This work demonstrates the importance of replication of predictive modeling experiments in education, and demonstrates how experimental design and modeling decisions can impact the extent to which model performance generalizes beyond the initial training data.

1. INTRODUCTION

The repeated verification of scientific findings is central to the construction of robust scientific knowledge, particularly in a fast-growing field such as educational data mining. This can take the form of (a) reproduction (reproducibility), using the original methods applied to the original data to reproduce the original results, and (b) replication (replicability), applying the original methods to *new* data to assess

the robustness and generalizability of the original findings. Since reproducibility is a necessary condition for replicability (an experimental procedure cannot be applied to new data if the procedure cannot be reproduced), achieving replicability requires solving the problem of reproducibility.

In this work, we discuss the reproducibility crisis in machine learning, noting specific challenges faced by applied researchers in the learning sciences, particularly in the subfields of educational data mining and learning analytics. We argue that existing frameworks for reproducible machine learning such as open code-sharing platforms and public code notebooks are valuable steps, but are insufficient to fully address the challenges both within our subfield of interest and the broader machine learning community. In particular, we argue that code-sharing does not address the breadth of challenges – experimental, methodological, and data – we face as practitioners, as Section 3 details. Instead, we propose a paradigm of *end-to-end* reproducibility for machine learning: fully reproducing (or replicating) the pipeline from raw data to model evaluation. End-to-end reproducibility is possible with current freely-available computing technologies, namely containerization.

Using an open-source platform for conducting reproducible end-to-end machine learning experiments on large-scale educational data, the MOOC Replication Framework (MORF), we conduct a three-stage replication experiment in Section 4, which is the primary contribution of this work.¹ Our case study evaluates both the experimental design, by comparing different train-test regimes, as well as the modeling, by replicating the original results and attempting to extend them via modern neural network regularization methods. We present practical recommendations based on our results in Section 5. We describe additional benefits, beyond reproducibility, afforded by MORF, and replication in machine learning more broadly, in Section 6, concluding in Section 7.

Josh Gardner, Yuming Yang, Ryan Baker and Christopher Brooks "Modeling and Experimental Design for MOOC Dropout Prediction: A Replication Perspective" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 49 - 58

¹The code to fully replicate these experiments, including their execution environments and software dependencies within a Docker environment, is available at <https://github.com/educational-technology-collective/dl-replication/>.

2. PRIOR WORK

2.1 The Reproducibility Crisis in Machine Learning

Much has been written about the reproducibility crisis in science, particularly in fields which conduct human subjects research such as social psychology. Recent empirical evidence has shown that issues with reproducibility are also widespread in the field of machine learning. A survey of 400 research papers from leading artificial intelligence venues shows that none of the works surveyed document all aspects necessary to fully reproduce the work; only 20-30% of the factors evaluated were adequately reported in the works surveyed [19]. A replication study of deep reinforcement learning algorithms [21] show that the variance inherent to statistical algorithms, the use of different hyperparameter settings, and even different random number generation seeds contribute to a lack of reproducibility in machine learning research and have a direct impact on whether experimental results and baseline model implementations replicate. A survey of 30 machine learning studies in text mining identified poor reproducibility due to lack of access to data, software environment, randomization control, and implementation methods [28]. None of the 30 works surveyed provided source code, and only one of 16 applicable studies provided an executable to be used to reproduce the experiments.

These reproducibility issues are partly attributable to culture and convention. A survey of authors published in the *Journal of Machine Learning Research* found that roughly one third intentionally did not make their implementations available, for reasons including a lack of professional incentives, a reluctance to publish messy code, and the convention that doing so is optional [33]. [30] observes that only five of 125 published articles in the journal *Biostatistics* have passed the (voluntary) reproducibility review since its inception two years prior, yet considers this effort “successful” compared to the reproducibility of previous work.

As big data and machine learning permeate disciplines, this crisis in replication has also affected other fields of study, including the learning sciences. This is especially relevant in cases of very large datasets where the majority of learning is computer-mediated, such as in Massive Open Online Courses (MOOCs). For example, [16] showed that a large-scale replication of machine learning models led to substantially different conclusions about the optimal modeling techniques for MOOC dropout, with several findings replicating significantly in the *opposite* direction of the original study (which was conducted on only a single MOOC). In an attempted replication of the “deep knowledge tracing” method originally introduced in [31], the results showed that much simpler methods could achieve equivalent performance, and that the performance gains demonstrated in the original work were at least partially due to data leakage [24]. Somewhat more favorably, in [2], the authors find that 12 of 15 experimental findings in MOOCs replicated using a production-rule framework, but an additional two findings replicated significantly in the *opposite* direction.

2.2 Existing Tools for Reproducible Machine Learning

An exhaustive survey of tools and platforms to support reproducible machine learning research is beyond the scope of this paper. However, we include a brief survey of tools most relevant to reproducible machine learning for predictive analytics in education.

OpenML [35] is “an open, organized, online ecosystem for machine learning” that allows users to create data science pipelines to address specific “tasks”, such as classification and clustering. The OpenAI Gym is an open-source interface for developing and comparing reinforcement learning algorithms [6]. Its wide use for both teaching and research serve as an example of how a subfield can create and adopt shared tools that meet researchers’ needs while enhancing reproducibility. Recently, several publishing platforms dedicated to reproducible computational research have also formed, such as ReScience ², CodaLab ³, and WholeTail [5]. These platforms unify code, data, computation, and presentation in a single location. CodaLab and WholeTail also use Docker containerization to ensure reproducibility.

Each of these platforms is an important step toward reproducible machine learning research, and many of them address key barriers. However, these tools are insufficient for many types of machine learning tasks, including supervised learning with large-scale behavioral data from MOOCs. In particular, none of these platforms supports replication where the underlying data sets are privacy-restricted and cannot be publicly shared. In some cases, such as WholeTail, the platform scope is explicitly limited to public (non-restricted) datasets [5]. However, in educational data, many of the types of unanonymizable data that are privacy-restricted are also necessary for analysis (such as the text of discussion forum postings, IP addresses, or student names). Such restrictions are also likely to drive away machine learning researchers from working with this data, as gaining access to unprocessed raw educational data can be difficult or impossible without close collaborators and strong institutional support. Even with institutional support, government regulations such as the Family Educational Rights and Privacy Act (FERPA) may restrict or complicate data sharing.

3. THE MOOC REPLICATION FRAMEWORK

The replication crisis is the result of a confluence of forces which must be collectively addressed in order to achieve end-to-end reproducibility. Prior work has identified three groups of challenges: experimental, methodological, and data challenges [17]. No existing solution discussed in Section 2.2 currently addresses all three barriers. In this section, we outline three key barriers to reproducibility, and describe how these barriers are addressed by the MOOC Replication Framework (MORF), the research tool used to conduct this experiment.

MORF itself is a Python toolkit, accompanied by a platform-

²<http://rescience.github.io/about/>

³<http://codalab.org/>

End-To-End Machine Learning

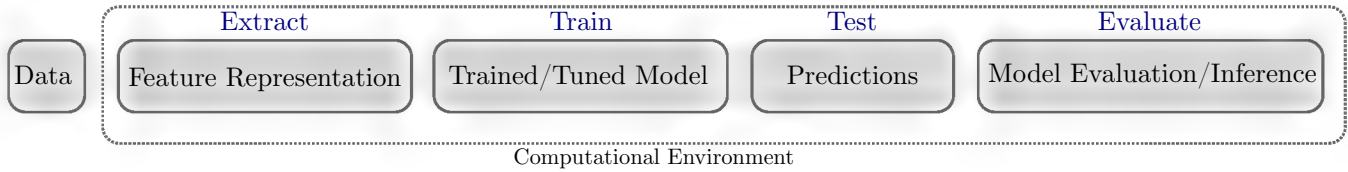


Figure 1: End-to-end reproducibility requires addressing data, technical, and methodological issues with reproducibility. Replication of the computational environment, in particular, is key to replicating this complete pipeline from raw data to results. Each of the four stages of the supervised learning pipeline executed inside the computational environment are encapsulated in MORF’s execution model (see Section 3).

as-a-service (the “MORF Platform”), which collectively address the challenges faced by researchers studying large-scale online learning data noted above [17].⁴

Users submit jobs to the MORF Platform using short, 4-5 line “controller” scripts which guide the execution of each stage of the end-to-end supervised learning pipeline (extract, train, test, and evaluate) shown in Figure 1. The use of controller scripts is a common approach to conducting reproducible computational research [25]. MORF’s combination of containerization and controller scripts allow the user to manage low-level experimental details (operating system and software dependencies, feature engineering methods, and statistical modeling) by constructing a Docker container which is submitted to MORF for execution. MORF manages high-level implementation details (parallelization, data wrangling, caching of results) by “bringing the computation to the data.” This prevents the download of sensitive raw data (currently, this includes the complete raw data exports from over 270 MOOCs offered by two institutions [17]). The containers used to execute each job are persisted in MORF’s public Docker Cloud repository, and the configuration file and controller scripts are persisted in Zenodo and assigned a unique Digital Object Identifier (DOI). This yields a reproducible end-to-end pipeline that is flexible, easy to use, and computationally efficient.

MORF eases the computational expense of conducting such research at scale by providing nearly an order of magnitude greater computational infrastructure than any of the platforms discussed in Section 2.2, and out-of-the-box parallelization to utilize it. See [17] for a more thorough comparison to other platforms.

3.1 Experimental Reproducibility via Containerization

Experimental challenges with reproducibility relate to reproducing the exact experimental protocol [17]. It has been noted that code-sharing alone is insufficient to guarantee reproducibility in computational research. For example, [10] showed that the published code accompanying 20% of their large sample of 613 published computer systems papers failed to build or run, and in total, it was not possible to verify or reproduce 75.1% of studies surveyed using the

artifacts provided in publication.

Even when code is available, other technical issues can prevent reproducibility in research workflows [25]. These include *code rot*, in which code becomes non-functional or its functionality changes as the underlying dependencies change over time (for example, an update to a data processing library which breaks backwards compatibility, or a modified implementation of an algorithm which changes experimental results), as well as *dependency hell*, in which configuring the software dependencies necessary to install or run code prevents successful execution [3]. This complex web of interdependencies is rarely described or documented in published machine learning and computational science work [19, 28], despite over two decades of evidence that it is a necessary condition for reproducing computational results [7].

MORF uses containerization to support end-to-end reproducibility (Figure 1). The Docker containers submitted to MORF fully encapsulate the code, software dependencies, and execution environment of an end-to-end machine learning experiment in a single file, ensuring end-to-end reproducibility and enabling sharing of the containerized experiment. Docker containers were developed to resolve many of the experimental reproducibility challenges described above in software development contexts [27], and are frequently used in industrial software applications, computational modeling, and computer systems research [3, 9, 23]. A major advantage of containerization over simple code-sharing is that containers fully reproduce the entire execution environment of the experiment, including code, software dependencies, and operating system libraries. Docker containers are more lightweight than a full virtual machine, but achieve the same level of reproducibility [27, 23]. Building Docker containers requires only a single Dockerfile (akin to a makefile) which contains instructions for building the environment. This imposes minimal additional burden on researchers relative to configuring, programming, and executing an experiment, but achieves a considerable increase in reproducibility. While other existing machine learning research platforms sometimes utilize Docker “under the hood,” this limits users’ ability to fully leverage containerization by configuring or sharing these environments. We are not aware of any platform which allows users to build and submit Docker images directly for execution as MORF does.

As part of MORF, we are assembling an open-source library of pre-built Docker containers to replicate experiments con-

⁴The MORF website, which includes documentation and short tutorials, is at <https://educational-technology-collective.github.io/morf/>

ducted on MORF to serve as shared baseline implementations. These containers can be loaded with a single line of code, allowing the research community to replicate, fork, interrogate, modify, and extend the results presented here⁵.

3.2 Methodological Reproducibility via Platform Architecture

Existing work on reproducibility largely focuses on strictly technical challenges, but as our experiment in Section 4 shows, methodological issues are at least as important. *Methodological challenges* to reproducibility reflect the methods of the study, such as its procedure for model tuning or statistical evaluation. Poor methodological decisions can lead to a lack of inferential reproducibility [19]. We see such issues in e.g. the use of biased model evaluation procedures [8, 36]; improperly-calibrated statistical tests for classifier comparison [12]; large-scale hypothesis testing where thousands of hypotheses or models are tested at once, such as in massive unreported searches of the hyperparameter space without statistical evaluation or appropriate corrections, or “random seed hacking,” wherein the random number generator itself is systematically searched in order to make a target model’s performance appear best or a baseline model worse [21].

MORF is designed to provide sensible default methodological procedures for many machine learning tasks, such as model evaluation, in practical terms nudging researchers to make sound choices. For example, MORF avoids the use of cross-validation for model evaluation: The prediction tasks to which most MOOC models aspire are prediction of *future* student performance (i.e., in an ongoing course where the true labels – such as whether a student will drop out – are unknown at the time of prediction). As such, using cross-validation within a MOOC session, when the outcome of interest is accuracy on a *future* MOOC session, provides an unrealistic and potentially misleading estimate of model performance. Prior work has demonstrated that within-session cross-validation in the MOOC domain can produce overly favorable estimates of classification performance on a future (unseen) course or future session from the same course [37, 4]. Adopting more effective model evaluation techniques by default requires no additional work for MORF users, and ensures that work produced on the MORF platform follows effective model evaluation procedures. MORF’s large data repository also prevents users from having to utilize only a single dataset for both training and testing; with many iterations of many unique MOOCs available, users can have considerable training data available while also conducting effectively-designed experiments with ample and representative test data.

3.3 Data Reproducibility via Execute-Against Access

Data reproducibility concerns the availability of data itself. In many domains, making raw data available is more an issue of convention than a true barrier to reproducibility. However, in the case of educational data mining, data are often governed by strict privacy regulations which protect the privacy of student education records. Similar restrictions af-

⁵The experiment presented below can be loaded by running `docker pull themorf/morf-public:fy2015-replication` in the terminal of any computer with Docker installed.

fect many other fields, from the health sciences to computational nuclear physics [25]. As a result, researchers are often legally prohibited from making their data available. Efforts such as [26] and [20] have attempted to address this problem in education by only releasing non-identifiable data, but many analyses require the original, unprocessed data for a full replication. Indeed, restricted data sharing is one of the main factors (in our experience) hindering generalizability analysis in educational data mining: investigators are generally limited to one or two courses worth of data (e.g. the courses they instruct or specific publicly available courses), and models are often overfit to these datasets.

MORF achieves data reproducibility while also meeting data privacy restrictions by providing strictly “execute-against” access to underlying data [17]. Most MOOCs are generated by a small number of platforms (e.g. Coursera, edX), and all courses from a given platform use publicly-documented data schemas, e.g. [11]. Thus, users can develop experiments using their own data from a given platform – or even the public documentation – and then submit these experiments for MORF to execute against *any* other course from that platform. This enables MORF to currently provide an interface to over 270 unique sessions of more than 70 unique courses offered by two different institutions on the Coursera platform, and to execute containerized experiments against this data in a secure, sandboxed environment by utilizing the shared public schema of the datasets [11]. These shared public data schemas also ensure that existing experiments in MORF can be replicated against new data (from the same MOOC platform) as it becomes available.

4. REPLICATION EXPERIMENT: NEURAL MOOC DROPOUT MODELS

In the remainder of this paper, we conduct an in-depth exploration of previously-published MOOC dropout prediction models using MORF. Neural models have demonstrated the capacity to achieve state-of-the-art performance on a wide variety of modeling and prediction tasks, from language modeling to computer vision. Their application to MOOC research has also demonstrated initial promising results [13, 29] due to their ability to model complex functional relationships between student behavior data and learning outcomes, but such research has been limited. In this section, we use MORF to replicate a comparison conducted in [13], which compares several machine learning algorithms using a set of seven activity features (e.g. number of lecture videos viewed, quizzes attempted, and discussion forum posts for each student) over each week in a MOOC in order to predict a binary dropout label indicating whether a user showed activity in the final week of a course.

This study is an ideal candidate for replication because it has been highly cited in the field, but compares six models over five weeks of a MOOC (effectively testing $\frac{6 \cdot 5 \cdot 5}{2} = 75$ pairwise hypotheses) using cross-validation on only a single dataset. This testing of many hypotheses/comparisons, with only a single observation for each, can lead to poor methodological reproducibility and provides no information about the variability of the estimates, relative to their magnitude [14]. Particularly because this experiment was concerned with empirical performance (in order to inform future “early warning” dropout prediction systems), obtaining an accurate

estimate of models' expected performance on *future* course sessions across a large, representative dataset can provide insight into the generalizability of these findings. The use of within-session cross-validation to draw conclusions about future generalization performance remains common in MOOC prediction research [15].

We present the results of our replication in Section 4.1, which matches the authors' original comparisons and their experimental design. The original work evaluated three different definitions of MOOC dropout; for tractability within this paper, we strictly replicate "Definition 1" of dropout from [13], which is the most commonly-used definition of dropout in MOOC research [16]. In Section 4.2, we present the results of a second experiment, which compares the same models using a different experimental setup (predicting on future course sessions) in order to demonstrate that the original experimental design overestimates the generalization performance of these models, likely due to overfitting. Our experimental analysis quantifies this overestimation and provides evidence using statistical tests. Finally, in Section 4.3, we use a comprehensive hyperparameter tuning and model evaluation procedure to show that we can improve the generalization performance of the LSTM model using batch normalization, but that the addition of two other neural network regularization methods does not further improve either the LSTM or RNN model. We also present detailed information on a variety of parameterizations examined in order to inform future work.

4.1 Experiment 1: Full Replication With Original Design

In our first experiment, we replicated the original experiment using the original design – estimating model performance via cross-validation within a single course – across 45 unique MOOCs using MORF, in consultation with the original authors. Results are shown in Figure 2.

The original work concluded that a Long Short-Term Memory (LSTM) neural network model “beats the ... other proposed methods by a large margin” [13], pp.1. – a result which matches the advances that neural models have made in other domains. Our results in Experiment 1 show, however, that (1) LSTM is actually one of the lowest-performing models, with the lowest average performance of any model tested in weeks 4 and 5; (2) in most cases, the 95% confidence intervals for algorithm performance overlap, and so we cannot conclude that there is a difference in performance between any but the very best and worst models evaluated, even without applying corrections to adjust for the use of multiple comparisons; and (3) observed performance of all models is lower than in [13], particularly in later weeks of the course.

We hypothesize that the relatively poor performance of LSTM may be due to overfitting on the dataset used in the original experiment in [13]. Particularly when using cross-validation for model selection on a single dataset with a highly flexible model such as LSTM, the experimental design of the original work was quite susceptible to overfitting. Overfitting seems particularly likely because no procedure for selecting hyperparameters was reported in [13], and some key hyperparameter settings for the LSTM model

(e.g. batch size) were not reported at all. These hyperparameters were not available even after correspondence with the authors, who did not record them and no longer had the original code available (which itself points to the need for reliable long-term reproducibility solutions such as MORF). The need for detailed hyperparameter reporting in reproducible research has been noted previously [21].

(2) shows the advantage of using MORF's large data repository, which allows us to observe variability in each algorithm's performance across many MOOCs to form confidence intervals for algorithm performance. Experiment 1 suggests that while differences in average performance may exist, these are too small to be interpreted as genuine and not spurious – particularly in light of the results shown in Figure 3, which shows that the differences due to cross-validation bias are larger than the observed differences between algorithms in most comparisons. We note that the out-of-fold prediction error in each cross-validation iteration could have been used to provide an estimate of the variability in model performance when applied to new data in [13]; however, this was not provided in the original work. In any case, having more datasets available makes the estimation of this variability more reliable than would have been possible with only one course.

Finally, the generally lower observed performance for all models may also be due to overfitting, and particularly due to the experimental design. Experiment 1 uses an identical design to [13]. However, [13] uses only a single course (in comparison to the 45 courses used in Experiment 1), which would have permitted tuning many of the hyperparameters germane to neural models (e.g. learning rate, activation functions, number of training epochs, batch size, number of hidden layers and units) specifically to optimize performance on this individual course. In contrast, when using the large, diverse set of courses in MORF, over-tuning such hyperparameters to an individual course would be disadvantageous (and extremely difficult, due to the diversity of course sizes and learner populations represented in MORF [17]). When fitting multiple courses at once (as in MORF), the incentive is instead to find a set of hyperparameters which generalizes to many different types of courses.

The results of this experiment demonstrate the importance of using large, diverse datasets for machine learning experiments, and of performing multiple experimental replications. Additionally, these results demonstrate that simpler models – such as RNN, radial SVM, and logistic regression – may achieve equivalent or better performance than LSTM for MOOC dropout prediction. Our results, which are contrary to the findings of the original study, also suggest that further replication is necessary to identify the most effective algorithms for MOOC dropout prediction, as we perform no hyperparameter tuning in Experiment 1 and only replicate the original models and features.

4.2 Experiment 2: Replication With Improved Experimental Design

The original aim of the dropout prediction model in [13] was to achieve accurate prediction on *future* courses. Prior work has shown considerable differences in prediction results depending on the prediction and transfer architectures

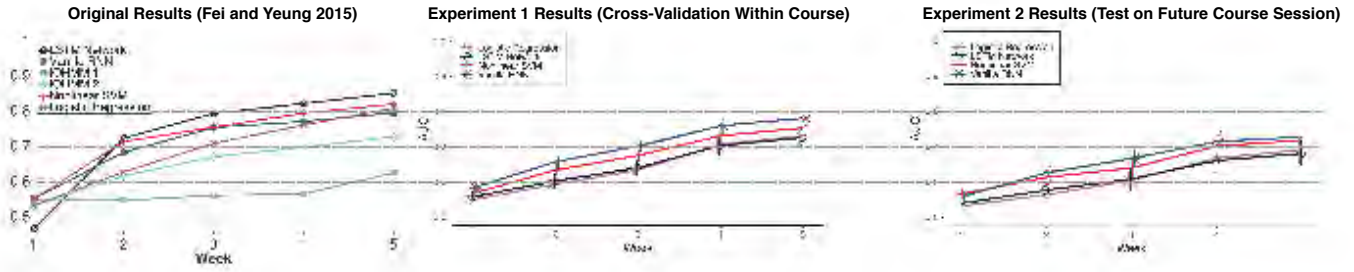


Figure 2: Original results from [13] (left) and replication results using the MOOC Replication Framework evaluated using a held-out future course session (center) and cross-validation (right). 95% confidence intervals shown. IOHMM not replicated due to lack of an open-source implementation which supported prediction.

used, particularly when predicting on the same course session used for model training instead of a future iteration of the same course [37, 4]. In the case of the original experiment, the cross-validation design was due to necessity: data from only a single MOOC was available to the original authors, and the prediction on out-of-fold data within the same session was used. However, MORF makes *all* sessions of each course available, with 45 courses having at least two sessions. Therefore, in this section, we conduct an experiment which uses the same modeling methods as the original experiment, but do so using a true holdout architecture, where each model is trained on the first $n - 1$ sessions of a MOOC, and is tested on the final session. We refer to this experiment as Experiment 2. Note that only the *design* of the prediction experiment is changed from Experiment 1.

The results of Experiment 2 are shown in the right panel of Figure 2. The contrast between the center (cross-validation) and right (holdout) panels demonstrates the optimistic bias which can be introduced by evaluating generalization performance via within-session cross-validation without the use of an independent hold-out session [36]. This matches previous results demonstrating that the bias of performance estimates when models are optimized over cross-validation folds can often exceed the difference between learning algorithms [8]. These results are further demonstrated by Figure 3, which shows a persistent positive bias for model evaluation performed by cross-validation versus the “true” performance on a future course session. A two-sided Wilcoxon signed-rank test of a null hypothesis of equivalence between the holdout and cross-validated experimental results, where each model-week combination was treated as an observation, was rejected with $p < 2.2 \times 10^{-16}$.

A notable result from Figure 3 is that the observed 2σ upper-bound on the bias due to cross-validation is around begins at roughly 0.035 AUC units when predicting after a single week, and increases to over 0.08 AUC points in later weeks. This difference due to design is as large as the difference between the highest-performing model and the two next-highest performing models in every week of the original experiment (see left panel of Figure 2). This shows that biases introduced by experimental design can entirely overshadow experimental effects, and should be a particularly strong call to action for the EDM community.

The findings of Experiment 2 are threefold. First, it demon-

strates that using within-course cross-validation to estimate generalization performance on future course sessions introduces a significant positive bias. This should serve both as a call for machine learning researchers to dedicate additional attention to the experimental design of machine learning experiments, as well as a call to practitioners to rigorously evaluate models prior to their deployment for prediction. Second, we quantify this bias, showing that in practice the bias roughly falls in the 95% CI $[0, 0.05]$ – where the upper-bound is larger than the difference between learning algorithms in the original work. Third, these results provide further evidence that the original results presented in [13] may have been overfit to the data in that work, and that the generalization performance of such a model may be lower than what was suggested by the initial results from [13].

4.3 Experiment 3: Improving Model Generalization Performance via Regularization

In the previous two experiments, we presented results which called into question some findings of the initial study in [13], that neural models can significantly improve MOOC dropout prediction. Our observations largely centered on the problematic potential for overfitting and optimistic bias due to experimental design in [13]. In this section, we show that the consequences of overfitting can be at least partially ameliorated by using modern regularization methods for neural models. This section is intended to explore (a) whether we can provide further evidence that the original models were overfit to the data in [13], and (b) whether we can improve the models’ generalization performance, and still achieve state-of-the-art dropout prediction performance with these models, through the use of regularization to reduce overfitting. In particular, we explore the recurrent neural network (“vanilla RNN”) and Long Short-Term Memory (LSTM) models from the original study, but introduce different architectures and explore various configurations of three regularization methods, none of which were in the original work. We show that we can approach, but not match, the performance reported in [13], even when predicting on future course sessions, by applying and tuning these regularizations to the RNN and LSTM models.

Regularization is a modification made to a learning algorithm that is intended to reduce its generalization error but not its training error [18]. We explore three types of regularization in this experiment, which we describe below.

Comparison of Cross-Validation vs. Holdout Experiment Results

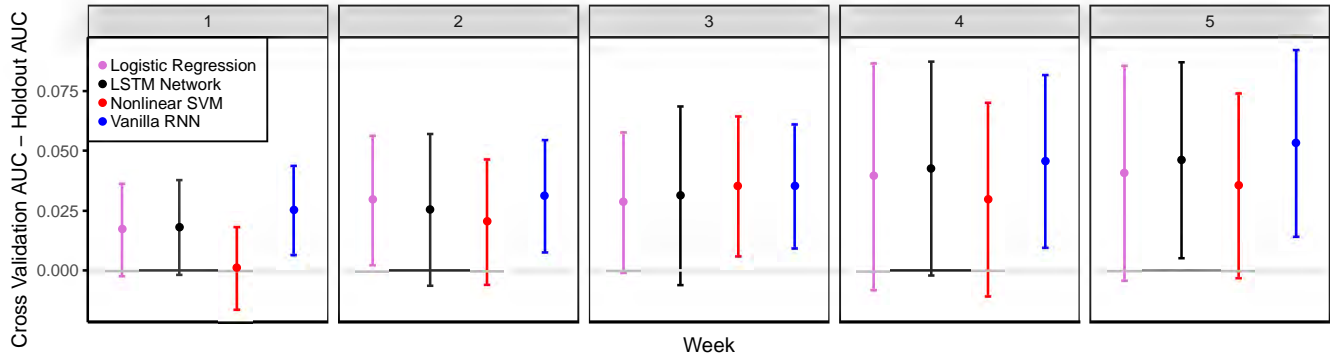


Figure 3: Comparison of AUC estimates from identical experiments using a holdout vs. within-course cross-validation evaluation architecture. These results show a persistent positive bias when within-course cross-validation is used to estimate predictive performance on a future, held-out session of a MOOC.

Dropout [34] randomly drops neural network units (along with their connections) during each training iteration with probability p . We test models with $p = \{0, 0.2\}$

Batch normalization [22] normalizes each feature in every minibatch of data during training, which stabilizes model training by reducing covariate shift (large changes in parameter updates due to differences in the distribution across features) and preventing small changes to the parameters from amplifying into larger and suboptimal changes in activations in gradients. We test models with and without batch normalization for each model.

L2 Regularization perhaps the most common regularization method in modern deep learning research [18], L2 regularization adds a penalty term to the loss function based on the L2 norm of the network weights with the parameter λ controlling the level of penalization (higher λ yields greater regularization). We test $\lambda = \{0, 0.01, 0.001\}$ for each model.

The study under replication was published in 2015, and since then, research on the regularization of neural models has advanced considerably. Dropout was only originally proposed in 2014, and batch normalization in 2015, so these techniques were still quite new at the time of publication of [13] and implementations were not widely available as part of standard neural network software, as they are now. As a result, the original publication quite reasonably did not explore these novel methods, despite their potential to improve their results in practice.

Experiment 3 evaluates dropout prediction after four weeks of the course (this is the final time point shown in Figure 2). Evaluating the full range of all weekly prediction tasks was beyond the scope of this experiment, as even the experiment here required testing 72 different hyperparameter configurations (36 LSTM and 36 RNN models) across 45 courses, resulting in a total of 3,240 total models trained. We choose the week 4 prediction task because, after four weeks, there would be maximal data for models to learn from – and also, potentially, for models to overfit to. Week 4 prediction therefore provides the best opportunity to separate models with strong generalization performance from

models which overfit.

Results from Experiment 3 are shown in Figure 4. We statistically evaluate the model comparisons, and visualize the results, using the Bayesian method of [14]. This method uses a hierarchical Bayesian model to evaluate all pairwise comparisons of a set of k models across N datasets by accounting for the correlation in performance across models and datasets. While the original comparison also accounts for fold-level correlation when applying the testing procedure to the results of a cross-validation procedure, here we only have a single estimate for each dataset, and this simply adds slightly more uncertainty to our estimation, which will tend to make the model more conservative (less likely to make a decision). The procedure estimates three probabilities for each pair of classifiers X and Y : $\mathbb{P}(X > Y)$, $\mathbb{P}(ROPE)$, $\mathbb{P}(X < Y)$, where ROPE indicates that the difference in performance between the models is within a “region of practical equivalence”, which in this experiment is set to $ROPE = 0.01$. We use a decision threshold of 0.9, which means that the procedure makes a “decision” (indicated by a colored entry in the windowpane plot of Figure 4) only when the posterior probability of one of the events is greater than or equal to 0.9; otherwise, the procedure makes no decision (indicated by a blank white entry in Figure 4).

The results in Figure 4 demonstrate that batch normalization considerably improves the LSTM model – all LSTM models with batch normalization performed better, on average, than any LSTM model without batch normalization. The hierarchical Bayesian model used to compare the modeling results across the 45 course in MORF indicated that almost all of the batch-normalized LSTM models were within the “region of practical equivalence”, or ROPE, indicating a high posterior probability that these models achieve practically equivalent predictive performance (in this experiment, a ROPE of 0.01 was used, meaning that a decision of ROPE indicates a confident decision that models’ test AUC differed by less than 0.01). Tuning of the other hyperparameters (L2 regularization λ , and dropout probability p) had little effect on observed model performance, with almost all hyperparameter configurations being practically equivalent within a fixed batch normalization group. These results suggest that

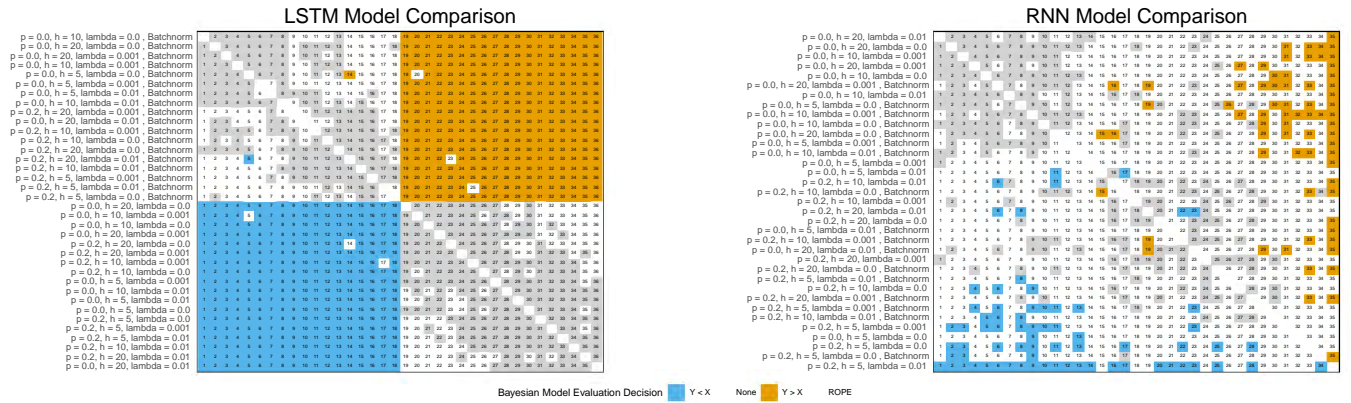


Figure 4: Windowpane plots for LSTM model tuning experiment (left) and RNN model tuning experiment (right). Batch normalization considerably improves the LSTM model. Other regularization methods (L2 regularization, dropout) show little effect. The RNN shows stable performance with a variety of configurations.

the LSTM model, which had considerably more parameters, was likely overfitting to the data – and that the different distributions across the different input features (which is countered by batch normalization) were a strong factor contributing to this overfitting. The LSTM model with the highest average performance, with $p = 0.0, h = 10, \lambda = 0.0$ and batch normalization, achieved a mean AUC of 0.726 on the MORF dataset – only 0.002 less than the best RNN model in Experiment 3.

Figure 4 shows a more complicated picture with respect to the RNN model. Generally, the results show that the RNN performance is more robust to the hyperparameters: the RNN achieves practically equivalent performance with a range of settings. For example, models with both 10 and 20 hidden units, and with every λ value considered, achieve performance practically equivalent to the highest-performing model. The exception to this robustness is dropout – every RNN in the “family” of best models (those with performance equivalent to the highest-performing model, as in [14]) had no dropout, or $p = 0.0$. The RNN model with the highest average performance, with $p = 0.0, h = 20, \lambda = 0.01$, achieved a mean AUC of 0.728.

Collectively, these results support the hypothesis that the LSTM model as parameterized in the original experiment tends to overfit on standard-sized MOOC datasets, while providing evidence that the RNN model is less prone to overfitting. The results also show that the LSTM model can be improved, with performance to match the best RNN model, through the use of regularization. For reference, the LSTM model in the original experiment (with a single layer and 20 hidden units) has 2,261 trainable parameters, while the RNN model has 581 tunable parameters. Regularization is less likely to impact a properly-parameterized model when sufficient data is provided, while its impact can be much more evident when a model has too many free parameters for the available data, as we see in the contrast between the LSTM and RNN results of Figure 4 and the lack of effect on RNN performance (mean AUC for the original RNN show in in Experiment 2 was 0.727 at week 5).

5. IMPLICATIONS FOR PRACTICE

5.1 Experimental Design for Predictive Modeling in EDM

The current work demonstrates that experimental design can have important implications for the conclusions generated from a machine learning experiment. Our results show that both researchers and practitioners should clearly identify the hypotheses to be tested by a machine learning experiment, or the goals of a deployed model, and then utilize experimental designs which allow for inference about the types of prediction scenarios to be encountered in the task of interest. Cross-validation can be useful in evaluating a specific dataset, e.g. by fitting an explanatory model where the interpretation of learned parameters is used to understand the dataset. Cross-validation may also be useful when the available data is limited. However, in cases where prediction on a future course or generalization to *new* data are of interest, using data from a future course or session will provide more reliable estimates of model performance. For further discussion of the design of machine learning experiments, see [1].

5.2 Hyperparameter Tuning for Neural Models in EDM

The current work provides evidence regarding the importance of effective model tuning and regularization in educational data mining. In practice, educational models need to operate effectively in a wide variety of scenarios. Predictive models in EDM are often required to obtain reasonable performance even as the dataset size, target population, and even data attributes change across course populations, institutions, or platforms, making model robustness a key consideration.

This experiment adds to an existing body of evidence (e.g. [14]) that selecting an appropriate statistical model can affect predictive performance much more than hyperparameter selection (when reasonable hyperparameters are selected). The introduction of regularization methods can improve models which are overfit, but has little impact on those which are not overfit. However, introducing regularization as an additional experimental factor may be particularly challenging

as it introduces an additional dimension of model tuning and can considerably increase the computational cost of experiments.

Our finding that batch normalization, in particular, improved the generalization performance of the LSTM model suggests that normalization of counting-based features – which can be highly skewed and show very different distributions for different types of actions – is an important tool for use in educational models.

6. BEYOND VERIFICATION: ADDITIONAL ADVANTAGES OF REPLICATION IN MACHINE LEARNING

Much prior work on reproducibility has focused on verification – ensuring that published results are true and can be reproduced. However, end-to-end reproducible machine learning frameworks, such as MORF, provide benefits beyond mere verification, including:

“Gold standard” benchmarking: open replication platforms allow for the comparison of results which were previously not comparable, having been conducted on different data. The use of such benchmarking datasets has contributed to the rapid advance of fields such as computer vision (e.g. MNIST, IMAGENET), natural language processing (Penn Tree Bank, Brown corpus), and computational neuroscience (openfMRI). These datasets have been particularly impactful in fields where it is difficult or expensive to collect, label, or share data (as is the case with MOOC data, due to legal restrictions on sharing and access). This can help advance the “state of the art” by providing a common performance reference which is currently missing in MOOC research.

Shared baseline implementations: We noted above that variability in so-called “baseline” or reference implementations of prior work has contributed to concerns about reproducibility in the field [21]. By providing fully-executable versions of existing experiments, MORF ameliorates these issues, allowing for all future work to properly compare to the exact previous implementation of a baseline method.

Forkability: containerization produces a ready-made executable which fully encompasses the code and execution environment of an experiment. These can be readily shared and “forked” much in the same way code is “forked” from a git repository. This allows MOOC researchers to build off of others’ work by modifying part or all of an end-to-end pipeline (for example, by experimenting with different statistical algorithms but using the same feature set as a previous experiment) within the same software ecosystem.

Generalizability analysis: Each successive replication of an experiment provides information about its generalizability. Evaluating the generalizability of experiments has been a challenge in MOOC research to date, where studies conducted on restricted and often homogeneous datasets are common [15]. When containerized end-to-end implementations are available, replicating these analyses on new data – even data which are not publicly available but share the schema of the original data – becomes as straightforward as

running the containerized experiment against new data.

Sensitivity Analysis: This technique, used widely in Bayesian analysis, evaluates how changes to the underlying assumptions or hyperparameters affect model fit and performance. Such an evaluation can provide useful information about a model’s robustness and potential to generalize to new data. Without being able to fully reproduce a model on the original data, sensitivity analyses are not possible. In MORF, such analyses can be conducted by simply forking and modifying the containerized version of the original experiment, then re-executing it against the same data. These analyses can also include so-called *ablation analyses*, wherein individual components are removed from a model to observe their contribution to the results, as well as *slicing analyses*, where fine-grained analysis of performance across different subgroups (e.g. demographic groups) is explored [32].

Full Pipeline Evaluation: Each stage of an end-to-end machine learning experiment (feature extraction, algorithm selection, model training, model evaluation) can be done in many different ways. Each stage also affects the others (for example, some algorithms might perform best with large feature spaces; others might perform poorly with many correlated features). However, current research usually evaluates only one or two components of this pipeline (e.g. training several algorithms and tuning their hyperparameters on a fixed feature set). Not only are the remaining stages often described in poor detail or not at all [19]; such work also leaves future researchers unable to evaluate the synergy between different aspects of the end-to-end pipeline in a published experiment (for example, exploring whether an algorithm’s performance improves with a different feature set). MORF fully encapsulates this end-to-end pipeline for a given experiment and it makes it available for modification to any other researcher.

Meta-Analysis: While meta-analyses are common in fields with robust empirical research bases, such analyses have been less common in the field of machine learning, which has an emphasis on novelty. The open availability of executable machine learning experiments affords detailed meta-analyses by providing complete results of all modeling stages for meta-analysis.

7. CONCLUSION

Further attention and analysis should be dedicated to replication in the field of educational data mining, and specifically to MOOC dropout prediction. This work proposes a paradigm of end-to-end reproducibility using the MOOC Replication Framework. Our case study in replication using MORF demonstrates the insights that can be gained from replication studies in EDM, and the importance of factors related to experimental design.

8. ACKNOWLEDGEMENTS

An earlier version of this work was presented at the Workshop on Reproducibility in Machine Learning at the 2018 International Conference on Machine Learning. We would like to thank the workshop organizers and participants for useful feedback on this work. We would also like to acknowledge the helpful assistance of the original authors of [13] in conducting this replication.

9. REFERENCES

- [1] E. Alpaydin. *Introduction to Machine Learning*. MIT Press, Aug. 2014.
- [2] J. M. L. Andres, R. S. Baker, G. Siemens, D. Gašević, and S. Crossley. Studying MOOC completion at scale using the MOOC replication framework. In *Proc. LAK*, pages 71–78, New York, Mar. 2018. ACM.
- [3] C. Boettiger. An introduction to docker for reproducible research. *Oper. Syst. Rev.*, 49(1):71–79, Jan. 2015.
- [4] S. Boyer and K. Veeramachaneni. Transfer learning for predictive models in massive open online courses. In *Proc. AIED*, pages 54–63. Springer, Cham, June 2015.
- [5] A. Brinckman et al. Computing environments for reproducibility: Capturing the “whole tale”. *Future Gener. Comput. Syst.*, Feb. 2018.
- [6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI gym. *arXiv preprint*, June 2016.
- [7] J. B. Buckheit and D. L. Donoho. WaveLab and reproducible research. In A. Antoniadis and G. Oppenheim, editors, *Wavelets and Statistics*, pages 55–81. Springer New York, New York, NY, 1995.
- [8] G. C. Cawley and N. L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *J. Mach. Learn. Res.*, 11(Jul):2079–2107, 2010.
- [9] J. Cito, V. Ferme, and H. C. Gall. Using docker containers to improve reproducibility in software and web engineering research. In *Web Engineering*, Lecture Notes in Computer Science, pages 609–612. Springer, Cham, June 2016.
- [10] C. Collberg, T. Proebsting, G. Moraila, A. Shankaran, Z. Shi, and A. M. Warren. Measuring reproducibility in computer systems research. Technical report, Univ. Arizona Dept. of Comp. Sci., 2014.
- [11] Coursera. *Coursera Data Export Procedures*. Coursera, June 2013.
- [12] T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comp.*, 10(7):1895–1923.
- [13] M. Fei and D. Y. Yeung. Temporal models for predicting student dropout in massive open online courses. In *Proc. ICDMW*, pages 256–263, 2015.
- [14] J. Gardner and C. Brooks. Evaluating predictive models of student success: Closing the methodological gap. *Journal of Learning Analytics*, 5(2):105–125, 2018.
- [15] J. Gardner and C. Brooks. Student success prediction in MOOCs. *User Modeling and User-Adapted Interaction*, 2018.
- [16] J. Gardner, C. Brooks, J. M. Andres, and R. Baker. Replicating MOOC predictive models at scale. In *Proc. ACM Learning@Scale*, 2018.
- [17] J. Gardner, C. Brooks, J. M. Andres, and R. S. Baker. MORF: A framework for predictive modeling and replication at scale with Privacy-Restricted MOOC data. In *IEEE Intl. Conf. on Big Data*, pages 3235–3244, Dec. 2018.
- [18] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press.
- [19] O. E. Gundersen and S. Kjetsmo. State of the art: Reproducibility in artificial intelligence. In *Proc. AAAI*, 2017.
- [20] M. A. HarvardX. HarvardX-MITx Person-Course academic year 2013 De-Identified dataset, version 2.0, May 2014.
- [21] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep reinforcement learning that matters. In *Proc. AAAI*, pages 3207–3214, 2018.
- [22] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. Feb. 2015.
- [23] D. M. Jacobsen and R. S. Canon. Contain this, unleashing docker for hpc. *Proc. Cray User Group*, 2015.
- [24] M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing? In *Proc. EDM*, pages 94–101, 2016.
- [25] J. Kitzes, D. Turek, and F. Deniz. *The Practice of Reproducible Research: Case Studies and Lessons from the Data-Intensive Sciences*. UC Press, Oct. 2017.
- [26] K. R. Koedinger, R. S. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the EDM community: The PSLC DataShop. *Handbook of educational data mining*, 43, 2010.
- [27] D. Merkel. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014(239), Mar. 2014.
- [28] B. K. Olorisade, P. Brereton, and P. Andras. Reproducibility in machine Learning-Based studies: An example of text mining. In *ICML Reproducibility in ML Workshop*, 2017.
- [29] Z. A. Pardos, S. Tang, D. Davis, and C. V. Le. Enabling Real-Time adaptivity in MOOCs with a personalized Next-Step recommendation framework. In *Proc. ACM Learning@Scale*, pages 23–32, 2017.
- [30] R. D. Peng. Reproducible research in computational science. *Science*, 334(6060):1226–1227, Dec. 2011.
- [31] C. Piech et al. Deep knowledge tracing. In *NIPS 28*, pages 505–513, 2015.
- [32] D. Sculley, J. Snoek, A. Wiltchko, and A. Rahimi. Winner’s curse? on pace, progress, and empirical rigor. In *ICLR 2018 Workshops*, Feb. 2018.
- [33] S. Sonnenburg et al. The need for open source software in machine learning. *J. Mach. Learn. Res.*, 8(Oct):2443–2466, 2007.
- [34] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.
- [35] J. N. van Rijn et al. OpenML: A collaborative science platform. In *Machine Learning and Knowledge Discovery in Databases*, pages 645–649. Springer Berlin Heidelberg, 2013.
- [36] S. Varma and R. Simon. Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, 7:91, Feb. 2006.
- [37] J. Whitehill, K. Mohan, D. Seaton, Y. Rosen, and D. Tingley. Delving deeper into MOOC student dropout prediction. *arXiv preprint*, Feb. 2017.

What will you do next?

A sequence analysis on the student transitions between online platforms in blended courses

Niki Gitinabard
North Carolina State
University
Raleigh, NC, USA
ngitina@ncsu.edu

Tiffany Barnes
North Carolina State
University
Raleigh, NC, USA
tmbarnes@ncsu.edu

Sarah Heckman
North Carolina State
University
Raleigh, NC, USA
sarah_heckman@ncsu.edu

Collin F. Lynch
North Carolina State
University
Raleigh, NC, USA
cflynch@ncsu.edu

ABSTRACT

Students' interactions with online tools can provide us with insights into their study and work habits. Prior research has shown that these habits, even as simple as the number of actions or the time spent on online platforms can distinguish between the higher performing students and low-performers. These habits are also often used to predict students' performance in classes. One key feature of these actions that is often overlooked is how and when the students transition between different online platforms. In this work, we study sequences of student transitions between online tools in blended courses and identify which habits make the most difference between the higher and lower performing groups. While our results showed that most of the time students focus on a single tool, we were able to find patterns in their transitions to differentiate high and low performing groups. These findings can help instructors to provide procedural guidance to the students, as well as to identify harmful habits and make timely interventions.

1. INTRODUCTION

Modern blended classrooms are defined by suites of educational tools such as learning management systems, online forums, intelligent textbooks, video lectures, groupware tools, and even ticketing systems for office hours. The ubiquity of such tools provides researchers with a rich amount of data on students' study behaviors, work habits, and their learning trajectories. This data can help researchers to identify good and bad study habits among students as well as to define measures for estimating students' performance early on in the courses. Large datasets of this type first became available in Massive Open Online Courses (MOOCs) that have supported informative research on students'

online study habits. While these tools have now become the norm in many classroom settings and while there has been substantial research on how students use the individual tools, we have far less understanding of how students work across tools and how different patterns of use may affect their learning. Our goal in this research is to address this question through the use of sequence mining. By developing a better understanding of student activities in different online systems and their transitions between these tools, we can provide the instructors with insight on how their students usually behave when they are not in class.

Prior research has shown that there are several features easily extracted from user logs that can distinguish high performing students from the lower performing ones. Researchers have found several informative features such as number of videos watched per week, completing assignments [27], starting early [28, 34], or skipping videos and assignments [12] that were associated with students' performance and dropout in MOOCs. Studies in blended courses showed that features such as course attendance, web page views, number of watched videos, number of pauses in videos, and the number of attempts before getting each question right are correlating with student dropouts [6].

More recent work in MOOCs, Intelligent Tutoring Systems (ITSs), and blended courses has focused on grouping the student activities into study sessions and analyzing these sessions and the sequence of students' actions in them. Some researchers have analyzed features based upon these sessions in MOOCs and blended courses, such as the duration [2, 29]. However, those studies overlook the patterns of student transitions between different states or different tools. Other researchers have studied the sequences of student actions in each session, but most of those studies are focused on MOOCs or ITSs and not many of them have focused on blended courses and the data collected from the several tools that the students use for these classes. Some of these studies have relied on Hidden Markov Models on the sequences of student actions and compared the diagrams between high and low performing students (e.g. [8, 10, 14]), while others have clustered these sequences to find groups of similarly behaving students in classes (e.g. [4, 11, 7, 17, 18, 19, 25, 30]). These studies have often been able to identify relevant clusters

Niki Gitinabard, Sarah Heckman, Tiffany Barnes and Collin Lynch "What will you do next? A Sequence Analysis of the Student Transitions between Online Platforms" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 59 - 68

among the students such as “confirmers” and “non-confirmers” [11] or “behind”, “on-track”, “auditing”, and “out” [17]. Also, other sets of studies have performed differential pattern mining on such sequences to find the patterns that are different between high and low performing students [16, 15, 13, 24]. And finally, another part of this research treats the sequences of actions as strings and uses analysis of N-grams to identify the popular trends in student activities and transitions [22, 5, 31, 32]. These methods are helpful in revealing many of the students’ behavioral patterns and the differences between the different performance groups, but are mostly focused on MOOCs or ITSSs.

Despite the extensive research in this area on MOOCs and ITSSs, studies on student transactions in blended courses are limited and most of them focus on correct/incorrect attempts on the same platform (e.g. the assignment submission systems) [11]. In this work, we collected activity logs from four online platforms for two offerings of two on-campus classes at North Carolina State University. In these classes, Piazza was used as a discussion forum, Moodle as a Learning Management System (LMS) was the means of sharing the course material and assignments, Github was used in one class as a version control as well as a code submission tool for the projects, and WebAssign was used for assignment submissions and automated grading in the other class. We aligned the logs into a single coherent transaction record, grouped the individual student actions into study sessions, and extracted the sequences of student actions from them. Finally, we labeled the students as the “Distinction” group who gained an A- or above and the “Non-distinction” group who gained a B+ or below in these courses and used N-gram analysis as well as Apriori studies to find the answers to the following research questions:

- RQ1** What are the most common transitions between different course tools?
- RQ2** Which transitions are significantly different between the distinction and non-distinction groups?

The answers to these questions can help us understand the trends of student activities better, to find key differences between high-performing students and the lower performing ones, and help the instructors to provide guidance to the students as they work or identify harmful patterns early in the semesters.

2. LITERATURE REVIEW

2.1 Students’ Online Activity Analysis

Since detailed online student logs have been available for the MOOCs, there have been extensive studies of student behaviors using these logs to identify their association with the students’ performance and attrition. Even simple measures such as number of videos watched are shown to be predictive of students’ attrition and performance in MOOCs. Some examples of these features include the number of videos watched per week, whether the student watched all of the lectures, or completed all of the assignments [27]. They also included joining the course early [28, 34], skipping videos or assignments, assignment performance [12], spending more time on each assignment [3], the number of lecture views/downloads, quiz attempts, and forum views/-posts/comments [9]. Some researchers such as Yang et al. have gone further and constructed more complex features to represent student confusion and shown that increased confusion is associated with dropout in MOOCs [33]. Chen et al. has studied

blended courses and has also shown that features such as course attendance, web page views, videos watched, video pauses, and assignment attempts are also correlated with student dropout [6]. All of these features, while informative, overlook an important part of the information that online logs provide us: the sequences of actions and transitions among different platforms.

To analyze a group of student actions as a whole, researchers have suggested defining study sessions. Prior work has suggested different methods for defining study sessions such as having a “fixed duration” [5], using “browser navigations”, or having a “cutoff” [2]. But as Kovanovic et al. showed, the choice of the method or the cutoff time is not trivial and there is no best method for everyone [20]. They suggested exploring the data to find the cutoff or method that matches the dataset best. Amnueypornsakul et al. defined study sessions and used the actions and the sessions to calculate measures such as the length of the action sequence, the number of occurrences of each activity, and the number of Wiki page views [2]. Sheshadri et al. also defined study sessions based on the time difference between student actions and extracted measures such as the average number of actions in each session, inconsistency of the student (i.e. how different the number of the sessions started by a student is from the class average and how infrequent they get online), average length of sessions, and sessions including discussion forum activity [29]. While these features can add to the information collected directly from the online tools, they still do not consider transitions from one type of action to the other.

2.2 Sequence Analysis

2.2.1 Markov Models

Several methods have been used for analyzing the sequences of student actions. The first and most popular is the use of Markov chains and Hidden Markov Models. Jeong et al. for example, trained models based upon system logs of a learning-by-teaching system called Betty’s Brain in which the students learn material by teaching an artificial agent, Betty [14]. The possible student actions in this platform are reading the material they are trying to teach Betty; editing the material; using links and concepts in forms of adding, removing, or changing (e.g. link add); querying the agent by asking questions about the provided material; asking Betty the agent to explain the answer she just gave; and giving a quiz to assess how well Betty has learned. The authors extracted sequences of student actions on the platform and used a Hidden Markov Model to analyze their behavior. They found that students who generated better concept maps used balanced learning strategies that include moving between different actions, while the students who generated low scoring concept maps typically focused too much on getting the quiz answers correct. Faucon et al. used semi-Markov chains to model student activities in 61 MOOCs offered by EPFL university on Coursera and EdX platforms [8]. They utilized an Expectation Maximization algorithm for fitting the model and showed a graphical representation of their results on the transitions between different states (e.g. submission, forum participation, video watching, etc) for students of different behavior profiles. Similarly, Geigle et al. used clickstream data from a UIUC Text Retrieval MOOC on Coursera to generate a transition diagram between the different tools [10]. While Markov models are suitable for modeling student transactions between different states and are easy to visualize, the differences they show are often hard to quantify and compare between groups [14].

2.2.2 Sequence Clustering

Another approach for analyzing sequences of student actions is by clustering them. Desmarais et al. for example, collected the action logs of students in a college math learning environment [7]. In that work, they defined distinct sessions where the students paused for more than 5 minutes between them unless the action after the pause was a submission to an exercise, which might take longer. They then clustered the sequences using the Levenshtein distance and identified three types of sessions. The first was when the students showed exploratory behavior and engaged in a mixture of browsing through exercises and notes. The second type were the short sessions comprising a variety of behaviors such as browsing and attempting the exercises and quizzes. The third were exercise intensive sessions mostly consisting of exercise logs. Kizilcec et al. used a similar approach on the student engagements in a MOOC [17]. For each assessment period, they labeled the students as either “behind”, “on track”, “auditing”, or “out” based on their engagement with the course material. Then, they applied K-means clustering on the sequences of the student states in all assessment periods to identify the prototypical engagement patterns and were able to observe four clusters of students as completing, auditing, disengaging, and sampling.

A similar analysis was performed by Guerra et al. on data collected from QuizJET, which was a voluntary practice platform for students in an introduction to programming blended course [11]. They extracted the sequence of correct and incorrect submissions for each student and each question. Then, by comparing the sequences of different students to the sequences of the same student, they observed that these sequences are personal and can show people’s study approaches like a “study genome”. While these genomes were shown to evolve throughout the semester, the evolved genomes for a single user were still more similar than the genomes across different users. They were able to cluster the students based on their genomes and identify two groups as the confirmers and the non-confirmers. The confirmers kept trying examples of the same topic even after they got one correct, while the non-confirmers moved on to the next topic after they were able to solve one example correctly. Finally, Boroujeni et al. clustered student activities in a MOOC and were able to identify four user profile types: users who watch videos before making submissions (44% of the users), users who make submissions without watching videos (2% of the users), users who watch videos and never submit (7% of the users), and the users who change their habit in the semester (47% of the users) [4]. These categories are similar to the ones suggested by Kizilcec et al. [17]. While clustering seems to offer much insight on similar sequences and differences between different groups of students, it is often challenging to interpret these clusters and get to real world groups of students.

To account for the randomness in the generation of Markov Models, some researchers have generated Markov Models based upon each individual sequence and then clustered them to obtain more meaningful results. Köck et al. for example, designed an analysis pipeline which included a pre-processor which extracted activity sequences from the raw data, a modelling unit which converted the sequences into Deep Markov Models, and a final clustering unit [19]. They applied this pipeline to extract common transitions exhibited by different performance groups in a Physics course at the US Naval Academy. Similarly, Shih et al. applied the same clustering method on the Hidden Markov Models based on student activities in a Geometry Cognitive Tutor

[30]. Klingler et al. developed an evolutionary clustering pipeline to improve cluster stability over multiple training sessions in the presence of noise [18]. This pipeline extracts action sequences from log data, transforms them into per-session Markov Chains, computes pairwise similarities between students for every session, then performs clustering using evolutionary clustering, and uses the Akaike information criterion with correction (AICc) to select the best model. They suggested that this pipeline can be used as a black box on any ITS. While the combination of clustering and Markov Models might overcome some disadvantages of each individual, the results are still challenging to interpret as noted by Shih et al. [30].

2.2.3 Sequences as N-grams

Another approach often taken when analyzing students’ sequence data is treating the sequence of actions as a sequence of strings, and then identifying the common N-grams in it. Li et al. and Sinha et al. for example, extracted the sequences of actions for users in MOOCs and used the frequency of N-grams in such sequences as predictive features to predict students’ performance and certification [22, 31]. Maldonado et al. also performed a similar analysis on data extracted from an interactive tabletop (Digital Mysteries) and were able to identify frequent sequences of actions that distinguish between different performance groups [23]. Wen and Rosè applied this method to extract the most common types of sessions among students and were able to identify 4 types of sessions as lecture and peer assessment sessions, browse course sessions, assignment and forum sessions, final quiz and survey sessions, and lecture and quiz sessions [32]. Brooks et al. defined fixed duration sessions during the semester (i.e. 1 day, 3 days, 1 week, and 1 month) and recorded students’ activity in each frame as a binary feature [5]. They used frequent N-grams extracted from these sequences as features to make early and cross-class predictions of student dropout. While N-grams are easier to process since there are many available libraries for analyzing them, extracting information from them can still be challenging and require expert help at times.

2.2.4 Differential Pattern Mining

A newer approach which is mostly applied to ITS data is Differential Pattern Mining. The algorithms in this approach are able to identify patterns that are more frequent than a specific threshold and are significantly different between the two specified groups such as pass/fail students [1]. Kinnebrew et al. for example used a differential sequence mining algorithm to extract the sequences that are different between the high performers and low performers using the Betty’s Brain platform [15, 16]. They found that the high performers more frequently engaged in reading activities in a monitoring context, while the lower performers usually perform short reads mostly not relevant to their recent actions [16]. Herold et al. applied the same analysis to the sequences collected with LivescribeTMDigital pens, used to complete all of their homework and exams [13]. These pens are able to log students’ handwriting as time-stamped pen strokes providing the sequence in which it was written. Using this method, they were able to identify 98 patterns in total and use them to make predictions on the students’ performance in the closest exam after the task with an R^2 of 0.3. While this approach is able to make the differences in performance groups bolder, it is still relatively new, the libraries for it are limited, and it is also possible to end up with a large number of rules that will need clustering again.

3. DATASET

We collected data from two offerings each of two distinct courses, a Discrete Math course (DM) in the Fall semesters of 2013 and 2015, and a Java programming course (Java) in the Fall of 2015 and 2016. The 2015 offerings of these courses occurred contemporaneously. Both of these courses are core undergraduate courses, required for students majoring and minoring in Computer Science. They both use significant online materials and support and can be considered blended courses. The online materials include online assignments, supplemental material, and student forums.

In all these classes, Moodle is used as an LMS for providing the course material and the assignment descriptions to the students. Piazza is used as the discussion forum and the main resource for the students in these courses to ask questions and get answers from the teaching staff as well as to have discussions with their peers. The students were able to post completely anonymously for a brief time in DM-2013 but it was blocked in all other courses. Posting anonymously to other students was always allowed. Posting on Piazza was not required in any of these classes, but it was encouraged by the teaching staff as the best choice of asking for help. In the DM classes, the instructors used multiple answer questions on WebAssign for a large portion of the assignments. WebAssign was configured to allow the students to attempt each question several times to get it correctly and provides the students with instant automatic feedback on their answers. In the Java classes, the students use Github as a version control for keeping track of their code and editing in teams, as well as the means for submitting their code for grading. The students' Github repositories were connected to Jenkins servers, which ran several test cases on their code after each pushed commit. Some of the tests were predefined and authored by the instructional staff and some others were the tests designed by the students to test their own code. This enabled students to get instant feedback on their code and possibly revise it after each submission. Our datasets in this study consist of the Piazza discussions, Moodle logs, and final grades for all the classes as well as Github commit logs for Java classes and WebAssign logs for DM-2013.

While some of the tools used in these classes are different, they play similar roles in the classes. In the DM classes students use WebAssign to submit their assignments and to receive immediate automated feedback. And in this class they can re-submit as many times as they wish to get the right answer. Similarly, in the Java classes the students use Github for making submissions on their projects. While these submissions often take more time than answering a simple question on WebAssign, the students are still able to get immediate feedback from Jenkins and to try again. Consequently, while some visible trends in these classes might be different, we expect the trends for WebAssign and Github to be similar, because they play a similar role. Similarly, in both these classes, Moodle and Piazza can be considered as support platforms since the students can use the course material, project descriptions, and the questions on the forum to resolve their confusions. The types of support these platforms are offering are quite different, since asking questions on Piazza is a more direct means of asking for help than referring to the class material.

More information on the population of these classes is shown in Table 1. The grade distributions for these classes are shown in Figure 1. Both these courses are C-wall courses, where the students need a C or better in them to proceed to the next computer science courses in the curriculum. As shown in these

Table 1: Statistics of Each Class

Class	DM-2013	DM-2015	Java-2015	Java-2016
Total Students	251	255	181	206
Teaching Assistants	5	5	9	9
Instructors	2	2	4	4
Average Grade	81.2	87.6	79.7	79.9

figures, most of the students performed well in these classes. Thus, we decided that clustering them into pass/fail groups would be uninformative and result in a skewed dataset. Since the median grades for all these datasets were close to 90, the cutoff between an A- and a B+ in the courses, we decided to partition the classes into two groups, the *distinction* group earning an A- or above, and the *non-distinction* earning a B+ or below. This partitioning resulted in an almost even groups of the students. We believe that this segmentation leaves room for adjusting the analysis for other classes with different grade distributions.

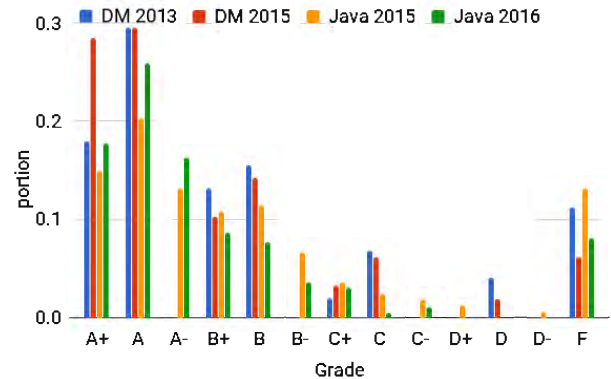


Figure 1: The Distribution of Grades in Different Classes

3.1 Discrete Math

This course covered material such as propositional logic, predicate calculus, methods of proof, elementary set theory, the analysis of algorithms, and the asymptotic growth of functions. The total enrollments in these classes consisted of 251 students in DM-2013 and 255 students in DM-2015. Both of these classes were offered in two sections by two instructors with 5 shared teaching assistants. The average final grade in DM-2013 was 81.2 and 87.6 in the 2015 class. Both sections in each year shared the same Moodle page for assignments and class material, a Piazza forum for discussions, and both used WebAssign as well as hand-graded assignments. The only major difference between these two offerings was that in 2015 the instructor consciously delayed responding to posts on Piazza so that the TAs and other students would be more involved. However, most of the posts were still answered in a similar time frame to the ones in 2013 by the lead TA in that class.

3.2 Java Programming Concepts

The material of the Java class mainly consisted of software design and testing, encapsulation, polymorphism, inheritance, linear data structures, finite-state machines, and recursion. The total enrollment in these classes was 181 students with an average grade of 79.7 in 2015 and 206 students with an average grade of 79.9 in 2016.

Both of these classes were offered in two different in-person sections by two separate instructors as well as a distance education section by two other instructors, having a total of four

instructors with nine shared teaching assistants. We removed the data for the distance education students from our analysis since they were a much smaller group and differed substantially from the local students who could engage in face-to-face interactions. These classes used Piazza for discussions, Moodle for sharing course materials, Github for working on group projects, and Jenkins for automated code evaluation.

While the teaching material and the methods were mostly similar across both the offerings, there was a major difference in the lab structures for these classes. Both course offerings included lab sessions. In each session, the students completed a short assignment in a team of three with assistance from the teaching staff. One key difference between the course offerings was in the structure of the lab sessions. In 2015, the labs were conducted in 8 class sessions, thus engaging all of the students and the TAs simultaneously. In 2016 however, students were enrolled in separate lab sessions (approximately 24 students each) with a dedicated TA and participated in 12 lab sessions. Additionally, in 2015, students continued to work with the same peers for all lab assignments while in 2016, they rotated partners after every four tasks, thus giving them a chance to meet and work with a wider variety of people.

4. METHODS

4.1 Action Sequence Generation

We began by collecting the logs from Piazza, Moodle, Github, and WebAssign for the courses. Later, we merged them into a single class-level transaction file sorted by time. We then generated study sessions on student activities based on their online transactions as discussed in our prior work [29].

As Kovanovic et al. suggested, we decided to explore our data to find the best method for generating study sessions [20]. Since there was no specific time length for the student sessions in our data, we decided to use a set cutoff time, m , for defining the sessions. If two consecutive actions are less than m minutes apart, they belong to the same study session. Otherwise, that session ends and the second activity after m minutes is a start of a new session. We plotted the average time differences between sessions, the total number of sessions, and the average number of activities per session for different cutoff times. These plots showed us two points with major changes that were chosen as the cutoff times for “study sessions” and “browser sessions”. We chose 15 minutes as the cutoff time for browser sessions, which show the times that the students have been online for the entire session. We also chose 40 minutes as the cutoff time for study sessions, which allows the time for the students to go offline for coding or solving problems on paper and get back online. We used this gap between online actions of the students considering that they often work offline before committing their code to Github or solve a problem on paper before submitting an answer on WebAssign. In this work, we focused on study sessions since they showed more transitions between different platforms. The total number of sessions for each group in each class is shown in Table 2. In the end, we recorded the sequence of student actions in these sessions for further analysis.

Table 2: The Total Number of Sessions for Distinction and Non-distinction Groups in Different Classes

Class Name	Count in Distinction	Count in Non-distinction
DM-2013	7,697	6,533
DM-2015	6,574	3,434
Java-2015	12,219	12,786
Java-2016	19,913	9,829

Similar to Kinnebrew et al. and Maldonado et al., we decided to compact the action sequences [16, 15, 23]. For that purpose, we replaced consecutive occurrences of the same actions by the “+” notion (e.g. MMM was replaced with M+). Our prior work showed that 90% of the student sessions consisted of access logs to the same platforms [29]. Also, the nature of most of these platforms requires consecutive submissions, such as multiple commits to Github for solving issues or multiple submissions on WebAssign until they find the right answer and there is not much of a difference between asking a question on Piazza after 5 submissions or 6. Abstracting these repetitions helps us spot the transitions between these platforms more easily and spot more similar sequences among students.

4.2 Sequence Mining

In order to explain our methods, we first need to define the common terminology in sequence mining. Based on Agrawal et al., the “support” of a sequence is defined as the ratio of occurrences of that sequence among all the sequences in the data [1]. For example, if a sequence S has happened 10 times among a student’s study sessions and the student has a total of 100 occurred sequences, the support for S will be 0.1 for that student. Looking at the support metric helps us to look into what percentage of this student’s sequences are S , rather than how many occurrences of S this student has. It also simplifies the comparisons between high and low performing students, since generally, the number of all actions for high performing users are higher and this might stop us from spotting the major differences between the students from different performance groups.

Another term often used in sequence mining is ‘confidence’. Based on Agrawal et al., the confidence of the action B following the action A ($A \rightarrow B$) shows how likely it is for action B to occur after A and is defined as:

$$Confidence(A \rightarrow B) = \frac{Support(A \cap B)}{Support(A)}$$

To identify the most common patterns among the students, we applied the idea of N-gram analysis as in prior work [22, 5, 31, 32, 23]. In text mining, an N-gram of length N (e.g. bigram) refers to a specific sequence of N words. Many times, the frequency or the count of N-grams are calculated and used as features. In this work, we treated the sequences of student actions as lists of words and using Scikit-learn library in Python [26], for each student, we calculated the support for all sequences of lengths of 2 - 3 to represent the transitions between every two tools and also keep room to count for repetitions. Then, we collected these numbers for the distinction and non-distinction groups into two separate lists for each sequence. We extracted the average support percentage for each sequence in each group to find the most common patterns among them. Additionally, we performed Kruskal-Wallis (KW) ANOVA test between the two lists for all sequences to find the patterns that occur with a different distribution among these

two groups [21]. The Kruskal-Wallis test is a good choice in this context because it does not assume normally-distributed data.

Also, to determine how likely the students are to transfer to a system after using another, we used the Apriori algorithm provided in Apyori library in Python. This algorithm is used to mine frequent item-sets and association rules [1]. It takes a minimum required support and performs in an incremental order, starting with single items (i.e. 1-sequences) that meet the support requirement (L_1) and add other items to the set as long as the support meets the criteria (L_k). In this work, we set the minimum support to a low number (0.02) to be able to find and compare even the rare transitions and the 1-sequences were defined as single actions on each platform. Based on Agarwal et al. the pseudo-code for this algorithm is as below:

```

 $L_1$  = frequent 1-sequences

for ( $k=2; L_{k-1} \neq \emptyset; k++$ ) do
     $C_k$  = New candidates generated from  $L_{k-1}$ 

    for All possible sequences c do
        Increment the count of all candidates in  $C_k$  that are
        contained in c
    end for
     $L_k$  = Candidates in  $C_k$  with minimum support
end for
Answer = Maximal Sequences in  $\bigcup_k L_k$ 

```

To find the transitions often associated together, we applied the Apriori algorithm on the sequences from distinction students and non-distinction students and calculated confidence for the frequent ones.

While participation on Piazza was not mandatory, it was strongly encouraged by the instructors as the primary venue for help seeking in all of the courses. As a result, we would expect to observe a large number of transitions between the submission tools (i.e. WebAssign and Github) and Piazza. We also expect these transitions to be more frequent after students make consecutive submission attempts since students who struggle with assignments often make several tries before contacting the instructors. We also expect higher-performing students to make more of such transitions because seeking help when they are struggling, rather than postponing it for later or going without, will help them to perform better in the course.

5. RESULTS

Since the tools used in these systems are different, we will present our results in each part for each class separately.

5.1 RQ1. What are the most common transitions between different course tools?

5.1.1 DM-2013

Our prior study on this class had shown that 90% or more of the student sessions are focused on a single tool and the sessions consisting of all WebAssign actions was the most common across them [29]. As Table 3, shows, consistent with our prior work, the most common sequence for both performance groups is repeated WebAssign Submissions, covering on average 70% of action sequences. This is not surprising due to the fact that the

students had unlimited submissions on this platform and often sought to “brute force” the answers.

The next most frequent pattern in both groups is multiple Moodle actions, which is again a unsurprising as students are required to log in on each session and must often navigate to their desired resources through a series of actions. Interestingly, transitions between WebAssign and Moodle are also comparatively frequent (the most frequent kind of transition between tools), consisting of approximately 4% of the total sequences. The more common transitions would be some submissions on WebAssign and moving to Moodle, while this sequence sometimes gets repeated several times as students move between these two tools and we can observe sequences like “w+m+w” on average in 0.4% of the students’ transitions or even more complicated ones such as “m+w+mw+”. Such transitions show students moving between class material like slides and the assignments and may show them referring to slides to revise their answers on WebAssign. We need to note that the sequences longer than 3 actions were not counted towards the calculation of support and confidence and thus, are not shown in the tables.

One would expect struggling students to move between WebAssign and Piazza to ask questions about the submissions, but as our results show, this transition does not happen frequently. Even among better performing students, it is more common to go to Moodle than Piazza after a couple of submissions, but it is even less likely for the lower-performing students. It seems like the students prefer to find the answers to their confusion among class material rather than asking questions or they prefer to leave help-seeking for another session.

Table 3: The Support for the Most Frequent Sequences in DM-2013 (W = WebAssign, M = Moodle, P = Piazza)

	Avg in Distinction	Avg in Non-distinction
W+	0.7064	0.7227
M+	0.1408	0.1615
W+M, M+W, MW, WM	0.0429	0.0380
P+	0.0303	0.0133
P+W, W+P, PW, WP	0.0039	0.0006
P+M, M+P, PM, MP	0.0004	0.0001

To better understand the student transitions between WebAssign and Moodle or WebAssign and Piazza, we calculated the confidence score for sequences in which Moodle and Piazza actions occur in the same session after one or more WebAssign actions. The results of the Apriori algorithm for this class are shown in Table 4. As we can see, there is almost a 10% chance of the students going to Moodle after one or more WebAssign submissions, while there is less than a 1% chance of them going to Piazza.

Table 4: Confidence for Different Transitions from WebAssign in DM-2013 (W = WebAssign, M = Moodle, P = Piazza)

	Distinction	Non-Distinction
$W \rightarrow M$	0.11	0.09
$W \rightarrow P$	0.007	0.003
$W+ \rightarrow M$	0.11	0.09
$W+ \rightarrow P$	0.007	0.003

5.1.2 DM-2015

The most frequent sequences for this class are shown in Table 5. Unfortunately, in this class, we do not have access to the WebAssign data. Thus, there were far fewer patterns found in this data than the 2013 class. But as with the prior offering,

the Piazza actions seem to be not nearly as common as Moodle actions. Additionally, the transitions between Moodle and Piazza were rare.

Table 5: The Support for the Most Frequent Sequences in DM-2015 (M = Moodle, P = Piazza)

	Avg in Distinction	Avg in Non-distinction
M+	0.913	0.966
P+	0.081	0.034
PM, MP, M+P, P+M	0.003	0.000

5.1.3 Java-2015, Java-2016

The most frequent sequences for these classes are shown in Table 6. These classes are similar to DM-2015 in that consequent Moodle actions is the most common sequence with an average about 55-65% of students' sequences in 2015 and about 70-80% of the student sequences in 2016. While in these classes Github commits are similar to WebAssign activities in DM-2013, the findings show that the students tend to commit their changes far less frequently than they submit questions on WebAssign. Multiple commits on Github are the next most frequent and they occur in about 20-30% of the student sequences in 2015 and 10-15% of sequences in 2016. Similar to DM-2013, where the students often moved between the submission system and the course material on Moodle, in these classes 4-6% of the student sequences are moving between Github and Moodle, where only 0.1-0.5% of the sequences refer to moving between Github and Piazza. In these classes also, moving back and forth a few times between the platforms is observed and we can see sequences such as "g+m+g+m" or "g+mg+m+g".

Table 6: The Support for the Most Frequent Sequences in Java Classes (G = Github, M = Moodle, P = Piazza)

	Avg in Distinction	Avg in Non-distinction
Java 2015		
M+	0.566	0.655
G+	0.294	0.204
G+M, M+G, GM, MG	0.041	0.043
P+	0.011	0.010
P+M, M+P, MP, PM	0.004	0.003
P+G, G+P, PG, GP	0.003	0.003
Java 2016		
M+	0.698	0.782
G+	0.134	0.089
G+M, M+G, GM, MG	0.062	0.052
P+	0.012	0.005
P+G, G+P, PG, GP	0.005	0.001
P+M, M+P, MP, PM	0.003	0.002

As with the DM-2013 class, we calculated the confidence score of action sequences that include Moodle and Piazza in the same session after one or more of Github actions. The results of the Apriori algorithm for these two classes are shown in Table 7. As we can see, there is a 28-37% chance of the students going to Moodle Github submissions, while there is only less than a 3% chance of them going to Piazza.

As our results show, the students seem more likely to go to the project descriptions or the course material after some submissions on Github rather than the discussion forum.

Table 7: Confidence for Different Transitions from Github in Java classes (G = Github, M = Moodle, P = Piazza)

	Distinction	Non-Distinction
Java 2015		
$G \rightarrow M$	0.31	0.36
$G \rightarrow P$	0.03	0.03
$G+ \rightarrow M$	0.31	0.37
$G+ \rightarrow P$	0.03	0.03
Java 2016		
$G \rightarrow M$	0.28	0.31
$G \rightarrow P$	0.02	0.007
$G+ \rightarrow M$	0.32	0.34
$G+ \rightarrow P$	0.02	0.01

5.2 RQ2. Which transitions are significantly different between the distinction and non-distinction groups?

5.2.1 DM-2013

The KW p-value results for the support percentages of different sequences in DM-2013 class is shown in Table 8. The significant values with $p < 0.05$ are marked as bold, while edge cases with $p < 0.1$ are marked in italics. We only included the significant and edge-case patterns and the transitions between platforms in the table. As these results show, the distinction students are significantly more likely to have a sequence of Piazza actions than the non-distinction group, with an average of 3% of their activities in the distinction group compared to 1% in the non-distinction group. The distinction students are also more likely to go to Piazza after a repetition of other activities than the non-distinction group. While the transition between WebAssign and Moodle (W+M, WM, M+W, MW) is high in both groups and not significantly different, the distinction group is more likely to move between Piazza and WebAssign (PW, WP, W+P, P+W) on average 0.4% compared to 0.01%.

Table 8: KW p-values between distinction and non-distinction students for different sequence supports in DM-2013 (W = WebAssign, M = Moodle, P = Piazza)

N-gram	Avg in Distinction	Avg in Non-distinction	KW pvalue
+P	0.0018	0.0001	3.31E-03
P-W transitions	0.0039	0.0006	3.08E-03
P+	0.0303	0.0133	1.30E-05
M-W transitions	0.0429	0.0380	0.644608

5.2.2 DM-2015

The KW p-values for the different sequences between the distinction and non-distinction group are shown in Table 9. Similar to the previous offering, the distinction group in this class is also more likely to have consequent Piazza activities, as well as go to Piazza after consequent actions on another platform. They are also more likely to move between Moodle and Piazza, while the non-distinction group is more likely to perform consequent actions on Moodle.

Table 9: KW p-values between distinction and non-distinction students for different sequence supports in DM-2015 (M = Moodle, P = Piazza)

N-gram	Avg in Distinction	Avg in Non-distinction	KW pvalue
P-M transitions	0.003	0	1.93E-03
+P	0.001	0	<i>5.82E-02</i>
M+	0.913	0.966	5.80E-05
P+	0.081	0.034	1.18E-04

5.2.3 Java-2015

The KW p-values for the different sequences between the distinction and non-distinction groups are shown in Table 10. Similar to the prior classes, the distinction group in this class was also more likely to go to Piazza after consequent actions on other platforms. Also, similar to DM-2015, the transitions between Moodle and Piazza are significantly more likely among the distinction group. Also, the distinction group has significantly more consequent actions on Github than the non-distinction group. However, while on average more sequences have a repetition of Piazza activities among distinction students, this difference is not significant in this class. Similarly, moving between Github and Moodle is more likely on average among the non-distinction group, but this difference is also not significant.

Table 10: KW p-values between distinction and non-distinction students for different sequence supports in Java-2015 (G = Github, M = Moodle, P = Piazza)

N-gram	Avg in Distinction	Avg in Non_distinction	KW pvalue
M+	0.566	0.655	0.046
P-M transitions	0.004	0.003	0.022
+P	0.003	0.002	<i>0.052</i>
P+	0.011	0.010	<i>0.095</i>
G+	0.294	0.204	0.005
G-M transitions	0.041	0.043	0.788
P-G transitions	0.003	0.003	0.336

5.2.4 Java-2016

The KW p-values for the different sequences between the distinction and non-distinction groups are shown in Table 11. Similar to the previous classes, in this class also we observe more repetitions of Piazza activities in the distinction group as well as more Piazza activities after a repetition of activities on other platforms. Also, similar to the 2015 Java offering and DM-2015, the non-distinction group is more likely to have consequent actions on Moodle. Despite the other classes, transitions between Github and Moodle as well as Github and Piazza are significantly different in this class and more likely for the distinction group. Comparing these results to the ones in Table 7, the findings seem conflicting since the non-distinction group is more likely to have Moodle activity in the same session after Github activities. However, we need to note that the Apriori algorithm, unlike N-grams, calculates the possibility of Moodle actions occurring after, but not necessarily consequently after, the Github activities. So, it seems like that the non-distinction group are more likely to move to Moodle at some point of the session after Github activities, but less likely to do so consequently after the Github actions.

Table 11: KW p-values between distinction and non-distinction students for different sequence supports in Java-2016 (G = Github, M = Moodle, P = Piazza)

N-gram	Avg in Distinction	Avg in Non_distinction	KW pvalue
M+	0.6976	0.7822	1.30E-05
+P	0.0021	0.0010	1.97E-02
P+	0.0123	0.0048	1.12E-03
G-M transitions	0.0616	0.0521	3.98E-02
P-G transitions	0.0046	0.0010	3.20E-04
P-M transitions	0.0026	0.0023	0.53

6. DISCUSSION

While the classes we analyzed and the offerings within them differ in topic, materials, structure, and instructor approach, our analysis shows that there are common patterns across all of them.

The first visible pattern is that the students are much more likely

to complete consecutive actions on the platform they are already using rather than switching to another platform. In all of the classes, the most common trend is two or more actions on WebAssign followed by Moodle in DM-2013, and Moodle followed by Github in the Java classes, while repetitions of Piazza actions seem to be more rare, even compared to platform switches. This might be due to the fact that most of the activities on Moodle, Github, and WebAssign consist of a sequence of smaller actions. For example, the students are much more likely to solve several problems on WebAssign or attempt a single problem several times, rather than only making a single attempt and leaving the platform. Similarly, on Moodle, the students often need more than one click to reach the material they need to access and on Github, the students are likely to push their code, face a failing test on Jenkins, and make a new commit to solve that issue. However, the actions on Piazza are not as closely monitored. On this platform, only making posts and replies are logged and viewing the posts or replies are not. Thus, the students are much more likely to make a single post or reply without any other visible actions on this platform and that might be a reason why consecutive Piazza actions are not as common as the other tools.

Another common pattern is that in contrast to our expectations, the students in all of the classes were much more likely to go back to the class material and the assignment descriptions on Moodle rather than rely on the discussion forum after one or more tries on their assignments. This was illustrated by the high confidence for transitions from WebAssign and Github (i.e. the submission systems) to Moodle (i.e. the indirect support platform), compared to transitions from these platforms to Piazza (i.e. the direct support platform), even in the higher performing students. As we expected, the visible trends for WebAssign and Github are similar in these classes due to the similarity in their educational role. As mentioned before, since the views are not monitored on Piazza, it is therefore possible that in some cases the students do refer to Piazza posts, only to find their answers in another student's question, without making any posts or replies. Thus, the lower amount of transitions to Piazza might be due to this difference in recording the activities. However, the teaching staff often found that the students did not look for their questions in their peers' posts and kept asking similar questions.

While both the performance groups have a large amount of consecutive Moodle actions, the non-distinction groups have on-average more of such sequences and this difference is often significant in these classes. Also, having repetitive Piazza actions and going back to Piazza after two or more actions on another platform is, on average, more common between the distinction students and this difference is significant in most of the classes, while in other classes an edge case that could be significant if we considered $p < 0.1$. This shows that while the non-distinction group seems to insist on finding the answer among the class material (or possibly reading the existing posts on Piazza), the distinction group seems to ask or answer questions on the discussion forum more often.

7. CONCLUSIONS

While multiple researchers have applied sequence analysis to educational data, most of this research has been focused on ITS data or MOOC data and there is not much research on the transitions of students between several resources in blended courses. In this study, we gathered logs from several online platforms that students interacted with in two offerings of two

undergraduate courses. We extracted sessions of studies among these activity logs and analyzed the sequences of the student actions in these sessions to find the general patterns in student transitions as well as the patterns that distinguish between the higher performing students and low-performers.

Our results show that consequent actions on the same platform are more likely for the students. Additionally, students are more likely to refer to the class material and the assignment descriptions rather than the discussion forum after a couple of submissions on assignments. However, the higher performers generally had more transitions between platforms and were often more likely to go to the discussion forum than the non-distinction group. We also found that even though some platforms used in classes are different, the results can be generalized across classes as long as the tools play similar educational roles, as WebAssign and Github did in our case. This can help findings to be expanded across a variety of courses using different platforms.

The results of this study can also help instructors identify helpful and harmful patterns among students and offer suggestions for forming more productive habits. The frequencies of these sequences added to the previously defined behavioral features can also help researchers improve the performance of their prediction models on student performances.

One limitation of this study is the differences between the length of the activities and how they are recorded on the different tools. Some types of activities are shorter and thus, more likely to repeat, such as WebAssign submissions where the questions are often multiple answers and quick to submit, while some other activities take a longer time, such as writing a Piazza post or solving an issue with the code and making a new commit. Additionally, while Moodle platform logs every action the users make online, Piazza only records the posts and replies and not the views. These differences in the tools might affect our findings. Further analysis, such as considering the time between actions differently for different tools might help us understand the trends in student activities better. Also, the WebAssign action logs are not available for the DM-2015 class, which limits the findings for this class and makes the comparisons between the two DM offerings less significant. Adding later similar offerings of these courses to the study in the future might help in finding more consistent trends.

In the future, we plan to expand the study to use different sequence analysis tools, such as the differential sequence mining tools. Those tools might be able to highlight other differences among the performance groups that are more difficult to spot using the current tools. Also, replicating our analysis on other courses and more offerings of the same courses can give us a better insight on how general some of these findings are. In the end, we plan on extracting predictive features from the student transitional patterns and add them to the other behavioral features to improve the accuracy of the performance prediction models on students, make the models fit better across classes, or make them fit better for earlier predictions in the semester.

8. ACKNOWLEDGEMENTS

This research was supported by NSF 1821475 “Concert: Coordinating Educational Interactions for Student Engagement” Collin F. Lynch, Tiffany Barnes, and Sarah Heckman (Co-PIs).

9. REFERENCES

- [1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993.
- [2] B. Amnueypornsakul, S. Bhat, and P. Chinprutthiwong. Predicting attrition along the way: The uiuc model. In *Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs*, pages 55–59, 2014.
- [3] J. M. L. Andres, R. S. Baker, G. Siemens, C. A. Spann, D. Gašević, and S. Crossley. Studying mooc completion at scale using the mooc replication framework. 2016.
- [4] M. S. Boroujeni and P. Dillenbourg. Discovery and temporal analysis of latent study patterns in mooc interaction sequences. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*, pages 206–215. ACM, 2018.
- [5] C. Brooks, C. Thompson, and S. Teasley. A time series interaction analysis method for building predictive models of learners using log data. In *Proceedings of the Fifth International Conference on Learning Analytics and Knowledge*, LAK ’15, pages 126–135, New York, NY, USA, 2015. ACM.
- [6] Y. Chen and M. Zhang. Mooc student dropout: Pattern and prevention. In *Proceedings of the ACM Turing 50th Celebration Conference - China*, ACM TUR-C ’17, pages 4:1–4:6, New York, NY, USA, 2017. ACM.
- [7] M. Desmarais and F. Lemieux. Clustering and visualizing study state sequences. In *Educational Data Mining 2013*, 2013.
- [8] L. Faucon, L. Kidzinski, and P. Dillenbourg. Semi-markov model for simulating mooc students. In *EDM*, pages 358–363, 2016.
- [9] M. Fei and D.-Y. Yeung. Temporal models for predicting student dropout in massive open online courses. In *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*, pages 256–263. IEEE, 2015.
- [10] C. Geigle and C. Zhai. Modeling mooc student behavior with two-layer hidden markov models. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, pages 205–208. ACM, 2017.
- [11] J. Guerra, S. Sahebi, Y.-R. Lin, and P. Brusilovsky. The problem solving genome: Analyzing sequential patterns of student work with parameterized exercises. In *Educational Data Mining conference*, July 2014.
- [12] S. Halawa, D. Greene, and J. Mitchell. Dropout prediction in moocs using learner activity features. *Experiences and best practices in and around MOOCs*, 7:3–12, 2014.
- [13] J. Herold, A. Zundel, and T. F. Stahovich. Mining meaningful patterns from students’ handwritten coursework. *Proceedings of EDM*, 2013.
- [14] H. Jeong and G. Biswas. Mining student behavior models in learning-by-teaching environments. In *Educational Data Mining 2008*, 2008.
- [15] J. S. Kinnebrew and G. Biswas. Identifying learning behaviors by contextualizing differential sequence mining with action features and performance evolution. *International Educational Data Mining Society*, 2012.
- [16] J. S. Kinnebrew, K. M. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students’ learning behavior patterns. *JEDM | Journal of Educational Data Mining*, 5(1):190–219, 2013.
- [17] R. F. Kizilcec, C. Piech, and E. Schneider. Deconstructing

- disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 170–179. ACM, 2013.
- [18] S. Klingler, T. Käser, B. Solenthaler, and M. H. Gross. Temporally coherent clustering of student data. In *EDM*, pages 102–109, 2016.
- [19] M. Köck and A. Paramythis. Activity sequence modelling and dynamic clustering for personalized e-learning. *User Modeling and User-Adapted Interaction*, 21(1-2):51–97, 2011.
- [20] V. Kovanovic, D. Gasevic, S. Dawson, S. Joksimovic, R. S. Baker, and M. Hatala. Penetrating the black box of time-on-task estimation. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge, LAK '15, Poughkeepsie, NY, USA, March 16-20, 2015*, pages 184–193, 2015.
- [21] W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.
- [22] X. Li, T. Wang, and H. Wang. Exploring n-gram features in clickstream data for mooc learning achievement prediction. In *International Conference on Database Systems for Advanced Applications*, pages 328–339. Springer, 2017.
- [23] R. M. Maldonado, K. Yacef, J. Kay, A. Kharrufa, and A. Al-Qaraghuli. Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop. In *Educational Data Mining 2011*, 2011.
- [24] P. Mukala, J. Buijs, and W. Van Der Aalst. Exploring students’ learning behaviour in moocs using process mining techniques. *Department of Mathematics and Computer Science, University of Technology, Eindhoven, The Netherlands*, pages 179–196, 2015.
- [25] N. Patel, C. Sellman, and D. Lomas. Mining frequent learning pathways from a large educational dataset. *arXiv preprint arXiv:1705.11125*, 2017.
- [26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [27] B. Pursel, L. Zhang, K. Jablokow, G. Choi, and D. Velegol. Understanding mooc students: Motivations and behaviours indicative of mooc completion. *J. Comp. Assist. Learn.*, 32(3):202–217, June 2016.
- [28] C. P. Rosé, R. Carlson, D. Yang, M. Wen, L. Resnick, P. Goldman, and J. Sherer. Social factors that contribute to attrition in moocs. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 197–198. ACM, 2014.
- [29] A. Sheshadri, N. Gitinabard, C. F. Lynch, T. Barnes, and S. Heckman. Predicting student performance based on on-line study habits: A study of blended courses. In *Proceedings of the 11th International Conference on Educational Data Mining (EDM) 2018, Buffalo, US*, pages 411–417, 2018.
- [30] B. Shih, K. R. Koedinger, and R. Scheines. Unsupervised discovery of student strategies. In *Educational Data Mining 2010*, 2010.
- [31] T. Sinha, N. Li, P. Jermann, and P. Dillenbourg. Capturing” attrition intensifying” structural traits from didactic interaction sequences of mooc learners. *arXiv preprint arXiv:1409.5887*, 2014.
- [32] M. Wen and C. P. Rosé. Identifying latent study habits by mining learner behavior patterns in massive open online courses. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, pages 1983–1986. ACM, 2014.
- [33] D. Yang, R. Kraut, and C. P. Rosé. Exploring the effect of student confusion in massive open online courses. *Journal of Educational Data Mining*, 8(1), 2016.
- [34] D. Yang, T. Sinha, D. Adamson, and C. P. Rosé. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Proceedings of the 2013 NIPS Data-driven education workshop*, volume 11, page 14, 2013.

Academic Performance Estimation with Attention-based Graph Convolutional Networks

Qian Hu

Department of Computer Science
George Mason University
Fairfax, Virginia
qhu3@gmu.edu

Huzefa Rangwala

Department of Computer Science
George Mason University
Fairfax, Virginia
rangwala@cs.gmu.edu

ABSTRACT

Student's academic performance prediction empowers educational technologies including academic trajectory and degree planning, course recommender systems, early warning and advising systems. Given a student's past data (such as grades in prior courses), the task of student's performance prediction is to predict a student's grades in future courses. Academic programs are structured in a way that prior courses lay the foundation for future courses. The knowledge required by courses is obtained by taking multiple prior courses, which exhibits complex relationships modeled by graph structures. Traditional methods for student's performance prediction usually neglect the underlying relationships between multiple courses; and how students acquire knowledge across them. In addition, traditional methods do not provide interpretation for predictions needed for decision making. In this work, we propose a novel attention-based graph convolutional networks model for student's performance prediction. We conduct extensive experiments on a real-world dataset obtained from a large public university. The experimental results show that our proposed model outperforms state-of-the-art approaches in terms of grade prediction. The proposed model also shows strong accuracy in identifying students who are at-risk of failing or dropping out so that timely intervention and feedback can be provided to the student.

Keywords

Educational data mining, graph convolutional networks, deep learning, attention

1. INTRODUCTION

Higher educational institutions face major challenges including timely graduation and retention of enrolled students. The National Center for Education Statistics (NCES) reports that the six-year graduation rate for first-time and full-time undergraduates is around 60%; the retention rate among first-time and full-time degree-seeking students is

around 80% [1]. These alarming statistics require higher educational institutions to take actions to improve their effectiveness and efficiency at educating students. Machine learning techniques have been increasingly developed and applied to educational settings in the hope of improving students' learning and increasing students' success [3, 26, 16]. Many systems and applications have been proposed; such as course recommender systems [7], academic trajectory and degree planning [20], educational early advising systems [9], and knowledge tracing for intelligent tutoring systems [35, 22]. Developing methods for accurate modeling and predicting students' performance is the key to these systems and applications.

Traditional performance prediction methods can be categorized into two types. The first builds a static model, which takes a feature vector as input (such as a student's grades in previous courses or student-related features) and outputs the predicted grades. A common approach that belongs to this category is linear regression methods [23]. Students take courses sequentially, i.e., they take some courses at each semester; and their performance in courses taken in the next semester depends on courses taken in previous semesters. Further, their knowledge evolves by taking a sequence of courses. To capture the temporal dynamics of students' knowledge evolution, sequential models have been proposed. A set of representative approaches within this category use recurrent neural networks (RNN) [12, 11].

Undergraduate degree programs are designed in a way that knowledge acquired in prior courses serves as prerequisites for future courses. The knowledge and skills required to do well in a course are acquired in multiple prior courses. The knowledge dependence between courses exhibit complex graph structure as shown in Figure 1. Figure 1 shows the prerequisite structures for computer science and civil and infrastructure engineering degree programs at George Mason University. Each node represents a particular course. An edge pointing from one course to another shows the prerequisite relationship. As an example, to do well in the data structure course (CS310), students need to acquire programming skills, object-oriented programming knowledge (CS211) and math (MATH113) which come from multiple different courses. The graph in Figure 1 also shows hierarchical relationships where a course can depend on another course which is at a much lower academic level. In addition to the prerequisite structures, degree programs are flexible, i.e., students can choose to take elective courses based on their interests and

Qian Hu and Huzefa Rangwala "Academic Performance Estimation with Attention-based Graph Convolutional Networks" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 69 - 78

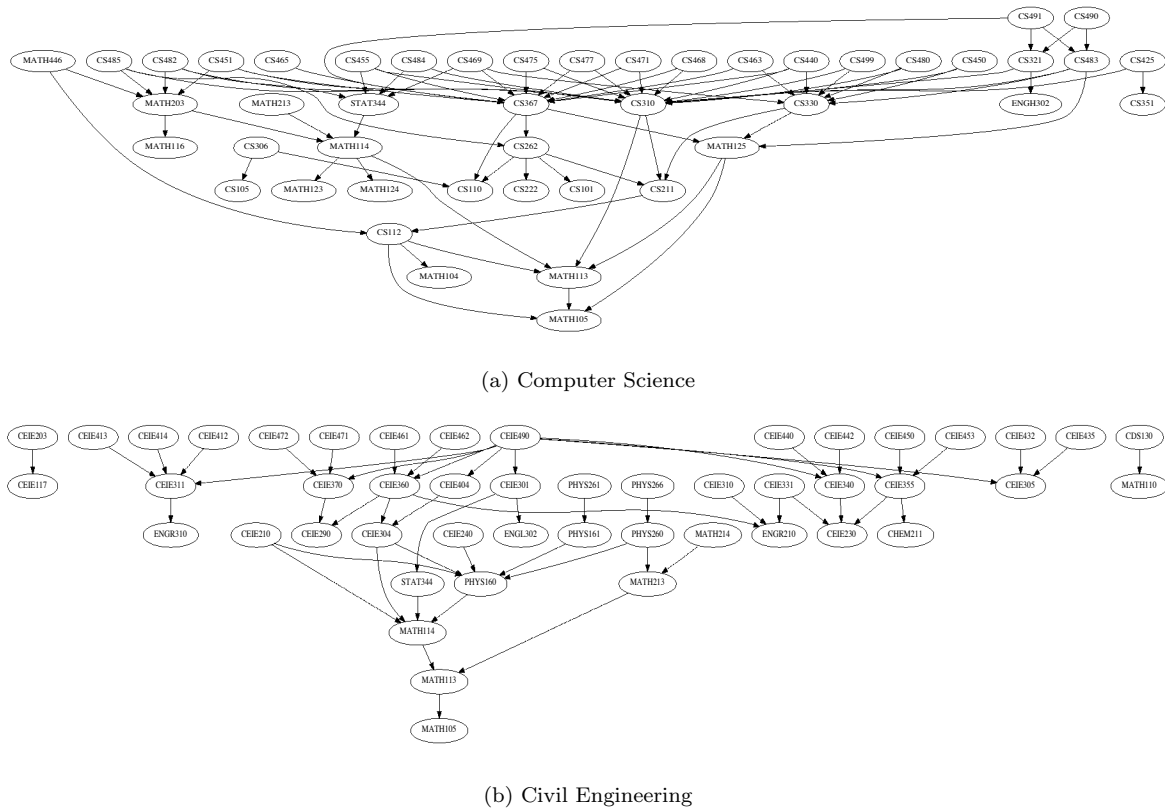


Figure 1: Course dependence structure in two representative majors.

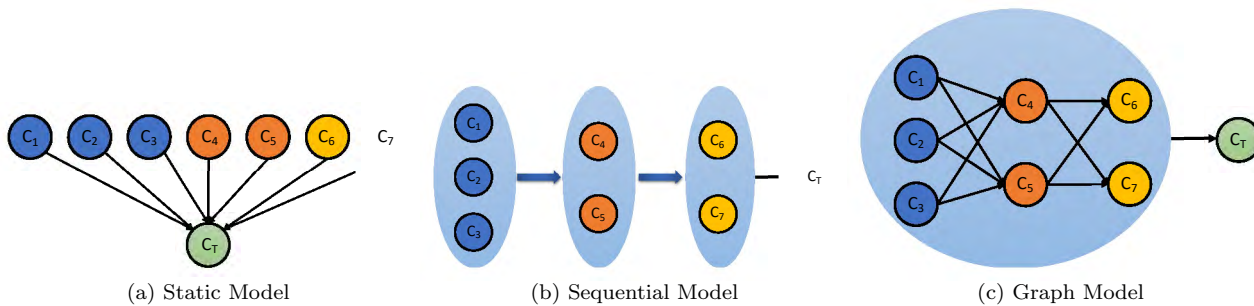


Figure 2: Comparison of Three Types of Model Architectures. In this example, a student takes course C_1, C_2, C_3 in the first semester, course C_4, C_5 in the second semester, course C_6, C_7 in the third semester before takes the target course C_T .

do not have to follow a specific ordering when taking these courses.

The complexity and flexibility of the degree programs make predicting students’ performance a challenge task. Prior approaches usually simplify or ignore these complex dependencies. Figure 2 shows the comparison of three types of models. Figure 2a shows a static model, where a student’s performance is directly dependent on a set of prior courses. Figure 2b shows a sequential model, where students’ knowledge evolution is partially modeled. To overcome the constraints and limitations of the traditional models, we propose a model based on graph convolutional networks to capture the complex graph-structured knowledge evolution exhibited by students’ data. Specifically, we propose an attention-

based graph convolutional network (ACGN) model for predicting a student’s grade in a future course. Figure 2c shows the graph model, where each course depends on all courses taken in the semester before it so that students’ knowledge evolution is fully captured.

When a system is used for decision making e.g., as a support tool for advisors to identify students who are at-risk of failing courses they will take; it is essential for the predictions to be interpretable. This allows the stakeholders to trust the decision making systems and make informed decisions. We show that our attention-based model is able to provide an interpretable and useful explanation for the predictions. Our model is able to analyze a student’s performance in prior courses and identify a collection of important prior courses

to explain the student’s performance in target course.

We performed extensive experiments on real-world datasets to evaluate our model and compare it with the other two types of models aforementioned. The experimental results are consistent with our observations that models with architectures more close to the degree program have better modeling capability and prediction performance. One of the important applications for students’ performance prediction is early warning and advising systems, where at-risk students are first identified and timely support is provided to improve their academic success. The experimental results show our model’s effectiveness at identifying at-risk students.

The key contributions of the paper are summarized as follows:

- Flexible graph structured model for students’ academic performance prediction. Observing the complex structures of undergraduate degree programs, we propose a graph convolutional network model for students’ performance prediction.
- Attention based model for explanation. Providing explanations for a model’s predictions makes the model useful for decision making. Our attention-based model can explain the predictions by identifying a set of prior courses important for the predictions.
- Identification of at-risk students. While most models achieve good performance at predicting students’ performance, they suffer from low accuracy at identifying at-risk students. Our proposed model is able to achieve comparable performance with state-of-the-art models.

2. RELATED WORK

The need to improve higher education services and offerings has attracted research on developing methods for predicting students’ performance [4, 27]. In this section, we review related work on students’ performance prediction. The related work can be classified into three categories: (i) static models, (ii) sequential models and (iii) graph models.

2.1 Static Models

Static grade prediction models learn a mapping function, where input is student-related features and the output is predicted grade. Polyzou et al. [23] proposed regression models specific to courses or students for predicting a student’s grade in a target course. They found that focusing on a course specific subset of the data leads to more accurate predictions. Elbadrawy et al. [8] introduced a personalized multi-regression model for predicting students’ performance in course activities. Compared to a single regression model, this model is able to capture personal student differences. To understand how students’ behavior impacts their academic performance, Wang et al. [32] collects students’ behavioral data using smart phone for performance prediction. Many other classic supervised learning approaches have been used for students’ performance prediction including decision trees [2], support vector machines and neural networks [31].

Adapted from recommender systems domain, matrix factorization [14] approaches are popular for grade prediction.

These factorization approaches make the assumption that a student’s knowledge/skills and a course’s knowledge components can be jointly represented with latent vectors (factors) [30]. Polyzou et al. [23] proposed course-specific matrix factorization models for grade prediction that decompose a course-specific subset of students’ grade data. The student course records also exhibit grouping structures and a domain-aware matrix factorization model was developed for the joint course recommendation and grade prediction [7]. Ren et al. [25] proposed matrix factorization model coupled with temporal dynamics for grade prediction.

2.2 Sequential Models

Students take courses sequentially. Their knowledge and skills evolve by taking a series of courses. To model the temporal dynamics of students’ knowledge evolution, sequential models have been proposed. Balakrishnan [5] proposed a Hidden Markov Model for predicting student dropout by modeling students’ activities over time in a Massive Open Online Courses (MOOCs). Swamy et al. [29] models student progress on coding assignments in large-scale computer science courses using recurrent neural networks. Kim et al. [12] proposed a bidirectional long short term memory (BLSTM) model for the online educational setting. Hu et al. proposed course-specific markovian models for students’ grade predictions [10]. Morsy et al. proposed cumulative knowledge-based regression models for next-term grade prediction, which models students’ knowledge evolution by using a sequential regression model. Hu et al. [11] proposed long short term memory models for grade prediction in traditional higher education.

2.3 Graph Neural Networks Models

Deep learning approaches have found unprecedented success in a myriad of applications involving regular structured data such as images (grids) and text (sequences) [18]. Graphs are more complex and irregular than grids or sequences and recent research efforts involve designing deep learning models for graph data. Graph neural networks have been proposed and applied to many areas such as computer vision for point clouds classification [33], action recognition [34]; recommender systems [6] and traffic prediction [19]. To the best of our knowledge, there is no prior work on students’ performance prediction using graph neural networks.

3. METHODS

3.1 Problem Statement

Given a student s , the set of courses taken and grades obtained in term t are represented by \mathbb{P}_s^t . For a sequence of terms $1 \dots T_s$, we denote $\mathbb{P}_s^{1 \sim T_s} = \mathbb{P}_s^1, \mathbb{P}_s^2, \dots, \mathbb{P}_s^{T_s}$ to represent the sequence of courses taken and grades obtained by student s in T_s terms. For a target course c taken in the future (next) term, the objective of the proposed method is to predict the grade student s will achieve in course c denoted by \hat{g}_s^c .

The proposed models are trained in a course specific manner i.e., for each target course c we learn a unique model. Due to the flexibility of academic degree programs, in each semester different courses can be taken; and for each student, the number of semesters studied before taking the tar-

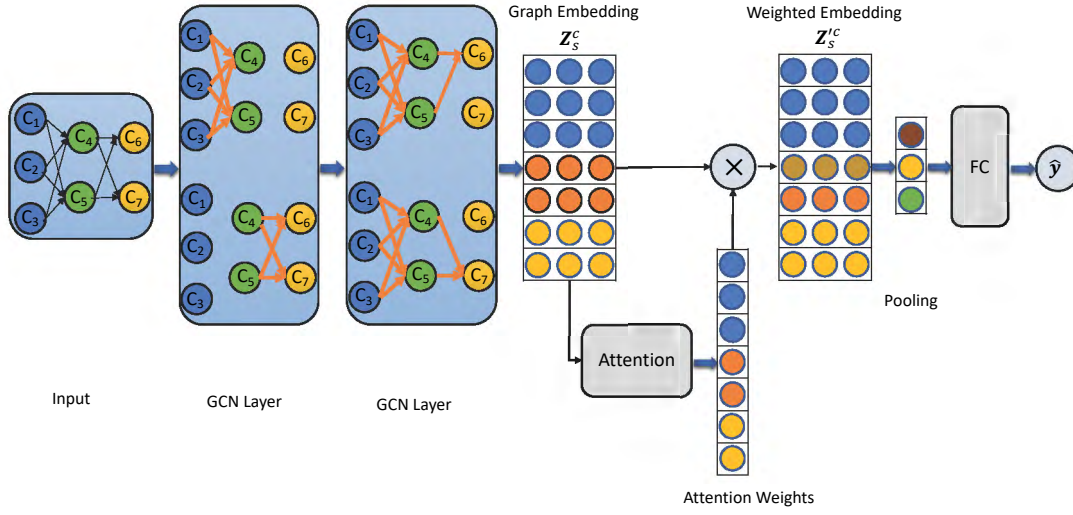


Figure 3: The proposed model.

get course will be different. Therefore, we index the length of the sequence with student-specific variable T_s .

For every target course c , a subset of frequently taken prior courses are identified from all the prior courses taken by students who have already taken the target course c . These prior courses are denoted as C_c of size N_c . For student s , only the prior courses in C_c are extracted from $\mathbb{P}_s^{1 \sim T_s}$ to form a graph which is represented by an adjacency matrix $\mathbf{A}_s^c \in \{1, 0\}^{N_c \times N_c}$ and a feature matrix $\mathbf{F}_s^c \in \mathbb{R}^{N_c \times D}$, where D represents the number of features. Take Figure 2c as an example, the student takes courses c_1, c_2, c_3 in the first term, c_4, c_5 in the second term and c_6, c_7 in the third term; we want to predict his/her grade in course C_T . Adjacency matrix \mathbf{A}_s^c for this student represents his course taken process. Courses taken in the current term are fully connected to courses taken in the next term; 1 represents connected, 0 otherwise. A row of the feature matrix \mathbf{F}_s^c represents the student's grades in corresponding prior courses.

3.2 Model Description

Figure 3 shows an overview of the proposed model. It is composed of three parts: 1) graph convolutional network, 2) attention layer and 3) a fully connected layer.

3.2.1 Graph Convolutional Network (GCN)

Convolutional neural networks (CNNs) show superior performance on several applications related to vision [15], speech and text [17]. CNNs are powerful because of their ability to exploit feature locality at multiple granularity. Graph Convolutional networks have a similar working mechanism but on data with more complex structures, namely, graph.

The input to a GCN is an adjacency matrix \mathbf{A}_s^c and feature matrix \mathbf{F}_s^c , encoding student s 's course taking process and grades in prior courses, respectively. Multiple layers of graph convolutional layer are applied on \mathbf{A}_s^c and \mathbf{F}_s^c to learn a graph level embedding $\mathbf{Z}_s^c \in \mathbb{R}^{N \times D}$. Each row of \mathbf{Z}_s^c corresponds to a node embedding vector. A graph convolutional layer is mathematically described as follows:

$$\mathbf{H}^{(l+1)} = f(\mathbf{H}^{(l)}, \mathbf{A}) = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}^{(l)}) \quad (1)$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ is the adjacency matrix with self-connections, $\tilde{\mathbf{D}} = \sum_j \tilde{\mathbf{A}}_j$ is the normalization matrix, $\mathbf{H}^{(l)}$ is the input and $\mathbf{W}^{(l)}$ is the weight matrix to be learned. $\mathbf{H}^{(0)} = \mathbf{F}_s^c$ and $\mathbf{H}^{(L)} = \mathbf{Z}_s^c$; namely, the input into the first GCN layer is the feature matrix \mathbf{F}_s^c , the output from the last GCN layer is the student-specific graph embedding \mathbf{Z}_s^c .

A filter in convolutional neural networks aggregates information from a pixel's neighbors. Similarly, the graph convolutional layer aggregates information from a node's neighboring nodes and generates a new node embedding vector by the following equation

$$\mathbf{h}_i = \sigma(\sum_j \frac{1}{c_{ij}} \mathbf{h}_j \mathbf{W}) \quad (2)$$

where node j is node i 's neighbor. A higher level of the node embeddings are generated by applying multiple GCN layers. Multiple layers of GCN aggregate information from a node's further neighbors. As shown in Figure 3, the first GCN layer aggregates information from a node's direct neighbors, namely, in our case the courses taken in last semester. The second layer collects information from a node's second degree neighbors, i.e., the courses taken two semesters ago. The final output is the graph embedding which entails information from all the courses a student has taken.

3.2.2 Attention Layer

The output from GCNs is a graph-level embedding matrix, which encodes information about a student's knowledge and skills acquired in prior courses. The knowledge acquired from different prior courses has different importance for the target course. To capture the importance differences of the prior courses, we integrate attention layer into our model. Attention mechanism allows the model to focus on the relevant features or information useful for prediction. It works by computing an importance score [24], higher score means the corresponding prior course is more important for pre-

dicting a student's performance; given by

$$e_i = MLP(\mathbf{h}_i) \quad (3)$$

$$\alpha_i = \frac{\exp(e_i)}{\sum_{k=1}^N \exp(e_k)} \quad (4)$$

where MLP is a learnable function, i.e., multi-layer perceptron, α_i is the attention score corresponds to \mathbf{h}_i . The output from the attention layer is an attention score vector α .

The graph embedding matrix \mathbf{Z}_s^c is weighted by attention scores to form a weighted graph embedding matrix $\mathbf{Z}_s'^c$ given by

$$\mathbf{Z}_s'^c = \begin{bmatrix} \alpha_1 \mathbf{z}_{s,1}^c \\ \vdots \\ \alpha_i \mathbf{z}_{s,i}^c \\ \vdots \\ \alpha_N \mathbf{z}_{s,N}^c \end{bmatrix} \quad (5)$$

Finally, the pooling layer coarsens the weighted graph embedding matrix into a latent vector \mathbf{v}_s^c . The latent vector is passed through a multilayer perceptron; the output from which is the predicted grade.

$$\hat{g}_s^c = f(\mathbf{v}_s^c) \quad (6)$$

where f is a multilayer perceptron network.

4. EXPERIMENTAL PROTOCOL

4.1 Dataset Description

Table 1: Dataset Statistics

Major	Fall 2017			Spring 2018		
	#S	#C	#G	#S	#C	#G
CS	5,042	16	47,889	5,297	20	52,152
ECE	1,992	18	34,355	1,980	18	34,170
BIOL	7,065	20	52,574	6,976	20	52,672
PSYC	5,367	20	25,207	5,368	20	25,247
CEIE	2,222	17	30,956	2,181	16	30,283
Overall	21,688	91	190,981	21,802	94	194,524

#S total number of students, #C number of courses for prediction, #G total number of grades

The data is collected at George Mason University from Fall 2009 to Spring 2018. The five largest majors are chosen including: 1) Computer Science (CS), 2) Electrical and Computer Engineering (ECE), 3) Biology (BIOL), 4) Psychology (PSYC) 5) Civil Engineering (CEIE). The evaluation procedure is designed in a way to simulate the real-world scenario of predicting the next-term grades. Specifically, the models are trained on the data up to term $T - 2$ and validated on term $T - 1$ and tested on term T . The latest two terms are chosen as testing terms, i.e. term Fall 2017 and term Spring 2018. For example, to evaluate the performance of the models on term Fall 2017, the model is trained on data from term Fall 2009 to term Fall 2016, validated on term Spring 2017 to choose the parameters associated with different approaches and finally tested in term Fall 2017. The statistics of the datasets are listed in Table 1

4.2 Evaluation Metrics

We evaluate the models from two perspectives: 1) the accuracy of grade predictions, 2) the models' ability at detecting at-risk students.

To evaluate the models' accuracy of grade prediction, two evaluation metrics are used a) mean absolute error (MAE) and b) percentage of tick accuracy (PTA).

$$MAE = \frac{\sum_{i=1}^N |g_i - \hat{g}_i|}{N} \quad (7)$$

where g_i is true grade and \hat{g}_i is predicted grade.

In the grading system, there are 11 letter grades (A+, A, A-, B+, B, B-, C+, C, C-, D, F) which correspond to (4, 4, 3.67, 3.33, 3, 2.67, 2.33, 2, 1.67, 1, 0). A tick is the difference between two consecutive letter grades. The performance of a model is estimated by how many ticks away the predicted grade is from the true grade. For example, the tick error between B and B is zero, B and B+ is one, B and A- is two. To use PTA for evaluation, we first convert the predicted numerical grade to its closest letter grade and then compute the percentage of errors with 0 tick, within 1 tick, and within 2 ticks denoted by PTA₀, PTA₁, and PTA₃, respectively.

We also evaluate the models' performance of identifying at-risk students. At-risk students are defined as those whose grades are lower than 2.0 (C, C-, D, F). The predicted grades below 2.0 are treated as positives and above 2.0 are treated as negatives. The process of detecting at-risk students is similar to grade prediction except that the output from the model (the predicted grade) is converted to 1 or 0 based on whether the predicted grade is below or above 2.0. As the number of at-risk students is low, we use F-1 score as evaluation metric.

4.3 Comparative Methods

Bias Only (BO)

Bias only method only takes into account a student's bias, a course's bias and global bias[23]. The predicted grade is as follow

$$\hat{g}_s^c = b^c + b_s^c + b_{c'}^c \quad (8)$$

where b^c , b_s^c , $b_{c'}^c$ are global bias, student bias and course bias, respectively.

Course Specific Matrix Factorization (CSMF)

The key assumption underlying this model is that students and courses can be jointly represented by low-dimensional latent factors. N , M and D is the number of students, courses and latent dimension, respectively [23]. To predict a student's grade in a course, we have:

$$\hat{g}_s^c = b^c + b_s^c + b_{c'}^c + \langle \mathbf{u}_s^c, \mathbf{v}_{c'}^c \rangle \quad (9)$$

where b^c is global bias, b_s^c is student bias term, $b_{c'}^c$ is course bias term; \mathbf{u}_s^c is student s 's latent vector, $\mathbf{v}_{c'}^c$ is course c 's latent vector.

Course Specific Regression (CSR)

Course specific regression (CSR) [23] is a linear regression model. The input into this model is a vector \mathbf{x}_s^c representing a student's grades in prior courses. A course specific subset

of prior courses included in $\mathbb{P}_s^{1 \sim T_s}$ are flattened to form the vector \mathbf{x}_s^c . The predicted grade is

$$\hat{g}_s^c = w_0^c + \mathbf{x}_s^c \mathbf{w}^c \quad (10)$$

where w_0^c is bias term and \mathbf{w}^c are weight vectors to be learned.

Multilayer Perceptron (MLP)

Multilayer Perceptron is a generalized version of CSR. CSR model is a linear model, which is not able to capture non-linear and complex patterns in students' grades data. Therefore, multilayer perceptron has been proposed by [11] for grade prediction. Similar to CSR, the input \mathbf{x}_s^c is a student's grades in prior courses.

$$\hat{g}_s^c = f(\mathbf{x}_s^c) \quad (11)$$

where f is the model to be learned.

Long Short Term Memory (LSTM)

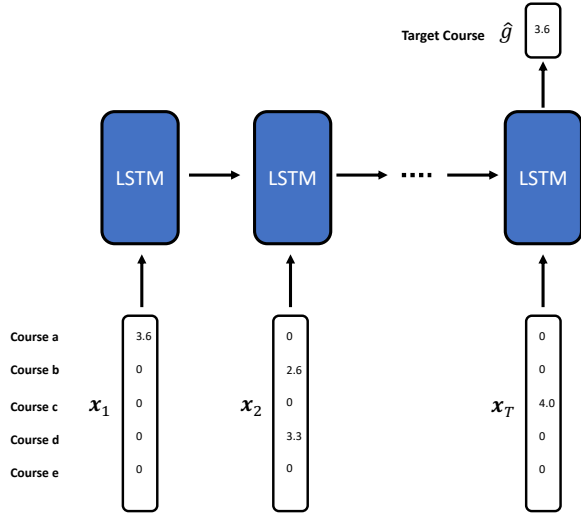


Figure 4: LSTM for grade prediction

Long short term memory (LSTM) is an extension of recurrent neural networks (RNN) for modeling sequential data. The assumption of using LSTM for students' performance prediction is that students knowledge and skills are evolving by taking courses in each semester. To capture the temporal dynamics of students' knowledge evolution, LSTMs have been proposed in [11]. The input $\mathbf{x}_{s,t}^c$ at time step t is a student's grades in courses at semester t . Many to one architecture is utilized and the output from the last step of LSTM is fed into a fully connected network; the output from which is the predicted grade. The model architecture is shown in Figure 4, where the courses a, b, c, d, e are prior courses, $\mathbf{x}_{s,t}^c$ encodes the student's grades in courses at time t and the output \hat{g} is the predicted grade.

4.4 Implementation

Our method is implemented in Pytorch [21]. For model optimization we use Adam [13]. To avoid model overfitting, we used l_2 norm regularization (with coefficient 0.001) and dropout (dropout rate 0.05) [28]. The number of dimensions for the graph embedding is chosen from a list of (8, 12, 16, 20, 32, 64).

5. EXPERIMENTAL RESULTS

5.1 Grade Prediction

Table 2 reports the performance of ACGN and comparative approaches for the task of next-term grade prediction for the Fall 2017 and Spring 2018 semesters using the MAE metric. The proposed ACGN model achieves the best performance in most cases except the Civil Engineering (CEIE) major. The CEIE major has relatively simpler knowledge dependence structure as shown in Figure 1b. A majority of higher level courses, such as 300 and 400 level courses for the CEIE major have shallow knowledge dependence. While for CS major, the higher level courses have deeper knowledge dependence or longer pre-requisite chains.

Another observation is that models which are able to capture the complex knowledge dependence more have better performance. The static models (BO, CSMF, CSR, MLP) are outperformed by sequential model (LSTM) in most cases, on average by 9.2%; the sequential model is outperformed by graph model (ACGN), besides CEIE major, on average by 7.0%. The experimental results are consistent with our assumption that the knowledge dependence in the undergraduate degree programs is complexly networked structures and a graph model is well-suited at capturing the underlying dynamics.

Table 3 shows the comparative performance using the percentage of tick error accuracy. In contrast to MAE, the PTA metric can provide a fine-grained view of the errors made by different methods. From Table 3 we observe that the performance gap between models at \mathbf{PTA}_0 is larger than at \mathbf{PTA}_2 . For example, for CS majors in Fall 2017, the gap between the best performing model ACGN and the worst performing model BO at \mathbf{PTA}_0 is 13.24%, which is larger than 8.53% at \mathbf{PTA}_2 .

5.2 Detecting At-risk Students

Detecting at-risk students early is a fundamental task for early warning and advising systems. We evaluate the models' performance at detecting at-risk students. Table 4 shows the experimental results evaluated by F-1 score. The percentages of at-risk students in different majors are presented at the table footnote. The PSYC major has the lowest percentage of at-risk students. The experimental results show that LSTM and ACGN achieve the best performance at detecting at-risk students. BO performs worst at the detection of at-risk students. BO only captures the average performance of a student and a course, which is biased by other students and courses' performance and the average performance of other students and courses is usually higher than 2.0 (the threshold of defining at-risk students).

5.3 Interpretation with Attention

Machine learning models have achieved impressive performance in many tasks. However, most of them remain black boxes and there are concerns about their transparency. A model's capability to provide explanations for its predictions can increase its transparency. For decision making, understanding the reasons behind predictions can help decision makers make informed decisions. Grade prediction models serve as an assistant tool for advisors to make decisions on whether to intervene on a student or not. When the model

Table 2: Comparative Performance of Different Models by MAE. (\downarrow is better)

Method	Fall 2017					Spring 2018				
	CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
BO	0.684	0.570	0.705	0.556	0.616	0.727	0.674	0.628	0.552	0.605
CSMF	0.594	0.476	0.550	0.517	0.479	0.647	0.539	0.499	0.492	0.491
CSR	0.607	0.444	0.551	0.440	0.441	0.628	0.493	0.463	0.439	0.444
MLP	0.585	0.390	0.515	0.407	0.413	0.590	0.436	0.417	0.413	0.369
LSTM	0.582	0.365	0.532	0.380	0.309	0.590	0.370	0.435	0.356	0.251
AGCN	0.540	0.335	0.459	0.309	0.336	0.543	0.366	0.379	0.316	0.258

Table 3: Comparative Performance of Different Models by Percentage of Tick Accuracy (\uparrow is better)

	Method	Fall 2017					Spring 2018				
		CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
PTA ₀	BO	16.76	20.75	14.40	15.52	14.90	16.07	12.11	15.42	13.65	19.79
	CSMF	20.00	23.58	22.40	23.10	28.85	22.31	17.37	23.35	28.41	28.65
	CSR	24.26	33.96	27.60	38.97	40.87	26.29	28.42	34.14	41.33	35.42
	MLP	26.32	39.62	31.00	41.72	41.35	27.76	33.68	41.41	43.17	42.19
	LSTM	27.21	42.92	37.40	48.62	49.52	30.54	54.74	42.73	49.82	57.29
	AGCN	30.00	41.51	38.80	56.21	50.00	36.52	39.47	44.49	50.55	56.77
PTA ₁	BO	44.71	49.06	43.20	57.59	48.56	44.09	37.37	46.70	57.20	50.00
	CSMF	55.15	62.26	60.00	63.10	62.98	52.72	54.74	63.66	59.04	61.98
	CSR	55.29	66.04	59.40	66.21	71.63	57.37	63.68	66.30	65.31	69.27
	MLP	56.91	69.81	62.80	69.66	74.52	60.03	68.42	68.28	69.37	76.04
	LSTM	58.24	73.11	61.40	73.79	79.33	59.10	72.11	72.03	75.65	82.81
	AGCN	62.21	75.47	70.00	77.93	79.81	63.61	77.89	74.89	77.86	84.90
PTA ₂	BO	72.94	81.13	72.40	84.83	81.25	73.97	74.21	77.75	87.45	79.17
	CSMF	80.00	86.79	83.60	83.45	87.50	75.30	84.21	84.36	85.24	84.38
	CSR	76.76	86.32	80.80	83.45	84.62	77.03	82.63	84.58	82.66	86.46
	MLP	79.85	89.62	82.80	85.86	86.54	79.42	86.32	86.34	84.13	90.62
	LSTM	77.35	86.79	79.20	84.83	90.87	77.69	83.16	84.58	89.67	91.67
	AGCN	81.47	92.45	85.60	88.62	91.83	80.21	88.95	87.67	88.93	93.23

Table 4: Predictive Performance at Identifying At-risk Students, F-1 Score (\uparrow is better)

Method	Fall 2017					Spring 2018				
	CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
BO	0.092	0.000	0.116	0.000	0.000	0.085	0.000	0.194	0.000	0.000
CSMF	0.385	0.415	0.585	0.154	0.429	0.349	0.291	0.620	0.364	0.526
CSR	0.398	0.514	0.649	0.438	0.490	0.500	0.543	0.623	0.429	0.450
MLP	0.383	0.426	0.630	0.438	0.500	0.534	0.472	0.676	0.400	0.605
LSTM	0.492	0.533	0.553	0.276	0.702	0.584	0.650	0.638	0.400	0.681
AGCN	0.516	0.500	0.660	0.438	0.615	0.594	0.571	0.685	0.483	0.550

The percentage of at-risk students for each major in Fall 2017 is CS (23.7%), ECE (18.9%), BIOL (25.8%), PSYC (8.3%), CEIE (15.9%); In Spring 2018, it is CS (23.7%), ECE (24.7%), BIOL (18.1%), PSYC (6.6%), CEIE (14.1%).

predicts that a student is at-risk of failing a course, knowing which prior courses results in the prediction can also help advisors provide personalized feedback to students.

Attention mechanism works by letting the model focus on important information for prediction. In our proposed model, the design of the attention layer lets the model focus on important prior courses. The output from the attention layer is a vector of scores representing the importance of the prior courses computed by Equation 4. In this section, we show by case studies how the attention scores from the attention

layer explain the model’s predictions, especially, why the model predicts that a student is at-risk of failing a target course.

Table 5 shows four case studies. We keep the most important prior courses identified by attention score. For the first case study, the target course is CS-310, the student’s true grade in the target course is F and the predicted grade is C-. The most important four courses identified by attention layer is MATH-212, MATH-125, CS-262, CS-211. The reason for predicting this student as at-risk is that the stu-

Table 5: Case Studies By Attention Score

Target Course	True Grade	Predicted Grade	Prior Courses	Grades	Attention Score
CS-310	F	C-	MATH-213	N	0.33
			MATH-125	N	0.33
			CS-262	N	0.33
			CS-211	C	0.01
CS-310	D	D	MATH-213	F	0.913
			MATH-114	C	0.072
			CS-211	N	0.015
BIOL-311	F	C	BIOL-213	C+	0.5315
			BIOL-214	C+	0.4685
BIOL-452	D	C	CHEM-211	C+	0.5271
			BIOL-214	B	0.2784
			BIOL-213	C	0.1945

N means that the student did not take the course. Courses in bold mean they are in prerequisites chain.

dent did not take MATH-212, MATH-125, CS-262, therefore lacks the necessary knowledge to do well in the target course. In the second case, the student's true grade in CS-310 is D, the predicted grade is D. The three most important courses are MATH-213, MATH-114, CS-211. The reason for predicting this student as failing the target course is that he failed MATH-213 and did not do well in MATH-114 and did not take CS-211, which is the prerequisite of the target course. In the third case, the student's true grade in the target course is F and the predicted grade is C. The two most important prior courses identified are BIOL-213 and BIOL-214, both are in prerequisite chain of the target course and the student did not do well in them. The fourth case shows that the student failed the target course BIOL-452 and the predicted grade is C. The three most influential prior courses are CHEM-211, BIOL-214, BIOL-213. Courses CHEM-211 and BIOL-213 are in prerequisite chain and the student did not perform well in them.

From the case studies, we can see that the attention layer identifies missing knowledge components for a target course, arising due to two reasons: 1) the student did not take some important prior courses, 2) the student did not do well in the corresponding prior courses.

5.4 Sensitivity Analysis

In this section, we evaluate the sensitivity of the model's performance with respect to the dimension of the graph embedding. In Figure 5, the x-axis is the embedding dimension and y-axis is MAE for Fall 2017 and Spring 2018 datasets. From Figure 5, we can see that the model's performance varies with the dimension size. Overall, its performance is quite stable across the different majors.

6. CONCLUSIONS

Students' performance prediction is a fundamental task in educational data mining. Predicting students' performance in undergraduate degree programs is a challenging task due to several reasons. First of all, undergraduate degree programs exhibit complex knowledge dependence structures. Secondly, undergraduate degree programs are flexible which means students can take courses without following specific order and they can choose to take whatever electives they are interested in. Traditional approaches like static and sequential models are not able to fully capture the complexity

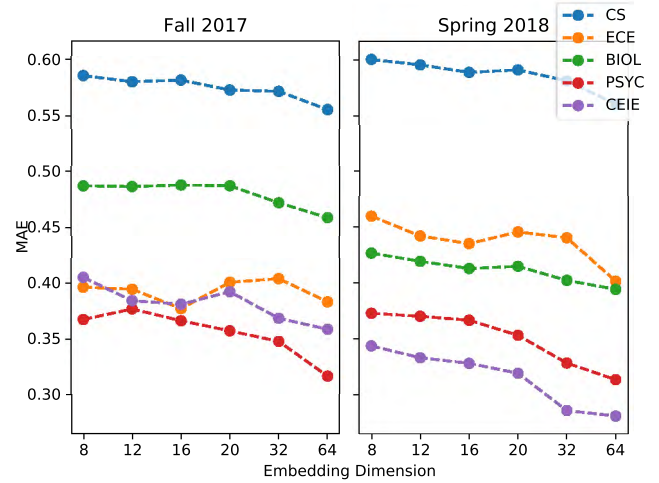


Figure 5: Sensitivity analysis on embedding dimension.

and flexibility of students' data.

In this work, we proposed a novel attention-based graph convolutional networks for students' performance prediction. The model is able to capture the relational structure underlying students' course records data. We performed extensive experiments to evaluate the proposed model on real-world datasets. The model is evaluated in several aspects: 1) grade prediction accuracy and 2) ability to detect at-risk students. The experimental results show that our model outperformed state-of-the-art approaches in terms of both grade prediction accuracy and at-risk students detection. Finally, the attention layer provides explanations for the model's prediction, which is essential for decision making.

7. ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation grant #1447489. The computational resources was provided by ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University, VA. (URL:<http://orc.gmu.edu>)

8. REFERENCES

- [1] Undergraduate Retention and Graduation Rates. https://nces.ed.gov/programs/coe/indicator_ctr.asp.
- [2] M. A. Al-Barrak and M. Al-Razgan. Predicting students final gpa using decision trees: a case study. *International Journal of Information and Education Technology*, 6(7):528, 2016.
- [3] R. S. Baker and K. Yacef. The state of educational data mining in 2009: A review and future visions. *JEDM| Journal of Educational Data Mining*, 1(1):3–17, 2009.
- [4] B. Bakhshinategh, O. R. Zaiane, S. ElAtia, and D. Ipperciel. Educational data mining applications and tasks: A survey of the last 10 years. *Education and Information Technologies*, 23(1):537–553, 2018.
- [5] G. Balakrishnan. Predicting student retention in massive open online courses using hidden markov models. Master’s thesis, EECS Department, University of California, Berkeley, May 2013.
- [6] R. v. d. Berg, T. N. Kipf, and M. Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.
- [7] A. Elbadrawy and G. Karypis. Domain-aware grade prediction and top-n course recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 183–190. ACM, 2016.
- [8] A. Elbadrawy, R. S. Studham, and G. Karypis. Collaborative multi-regression models for predicting students’ performance in course activities. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 103–107. ACM, 2015.
- [9] Q. Hu, A. Polyzou, G. Karypis, and H. Rangwala. Enriching course-specific regression models with content features for grade prediction. In *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 504–513. IEEE, 2017.
- [10] Q. Hu and H. Rangwala. Course-specific markovian models for grade prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 29–41. Springer, 2018.
- [11] Q. Hu and H. Rangwala. Reliable deep grade prediction with uncertainty estimation. *arXiv:1902.10213*, 2019.
- [12] B.-H. Kim, E. Vizitei, and V. Ganapathi. Gritnet: Student performance prediction with deep learning. *arXiv preprint arXiv:1804.07405*, 2018.
- [13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [14] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.
- [15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [16] C. Lang, G. Siemens, A. Wise, and D. Gasevic. *Handbook of learning analytics*. SOLAR, Society for Learning Analytics and Research, 2017.
- [17] Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.
- [18] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [19] Y. Li, R. Yu, C. Shahabi, and Y. Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.
- [20] S. Morsy and G. Karypis. A study on curriculum planning and its relationship with graduation gpa and time to degree. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 26–35. ACM, 2019.
- [21] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
- [22] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Advances in neural information processing systems*, pages 505–513, 2015.
- [23] A. Polyzou and G. Karypis. Grade prediction with models specific to students and courses. *International Journal of Data Science and Analytics*, 2(3-4):159–171, 2016.
- [24] C. Raffel and D. P. Ellis. Feed-forward networks with attention can solve some long-term memory problems. *arXiv preprint arXiv:1512.08756*, 2015.
- [25] Z. Ren, X. Ning, and H. Rangwala. Grade prediction with temporal course-wise influence. *arXiv preprint arXiv:1709.05433*, 2017.
- [26] C. Romero, S. Ventura, M. Pechenizkiy, and R. S. Baker. *Handbook of educational data mining*. CRC press, 2010.
- [27] A. M. Shahiri, W. Husain, et al. A review on predicting student’s performance using data mining techniques. *Procedia Computer Science*, 72:414–422, 2015.
- [28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
- [29] V. Swamy, A. Guo, S. Lau, W. Wu, M. Wu, Z. Pardos, and D. Culler. Deep knowledge tracing for free-form student code progression. In *International Conference on Artificial Intelligence in Education*, pages 348–352. Springer, 2018.
- [30] M. Sweeney, H. Rangwala, J. Lester, and A. Johri. Next-term student performance prediction: A recommender systems approach. *arXiv preprint arXiv:1604.01840*, 2016.
- [31] S. Umair and M. M. Sharif. Predicting students grades using artificial neural networks and support vector machine. In *Encyclopedia of Information Science and Technology, Fourth Edition*, pages 5169–5182. IGI Global, 2018.
- [32] R. Wang, G. Harari, P. Hao, X. Zhou, and A. T. Campbell. Smartgpa: how smartphones can assess and predict academic performance of college students. In *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing*,

- pages 295–306. ACM, 2015.
- [33] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.
 - [34] S. Yan, Y. Xiong, and D. Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
 - [35] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*, pages 171–180. Springer, 2013.

Evaluating Fairness and Generalizability in Models Predicting On-Time Graduation from College Applications

Stephen Hutt
University of Colorado Boulder
594 UCB, Boulder,
Colorado, 80309, USA
stephen.hutt@colorado.edu

Margo Gardner
University of Colorado Boulder
594 UCB, Boulder,
Colorado, 80309, USA
margo.gardner@gmail.com

Angela L. Duckworth
Character Lab & University of Pennsylvania
3401 Market St. Philadelphia,
PA 19104 USA
aduckworth@characterlab.org

Sidney K. D'Mello
University of Colorado Boulder
594 UCB, Boulder,
Colorado, 80309, USA
sidney.dmello@colorado.edu

ABSTRACT

We explore generalizability and fairness across sociodemographic groups for predicting on-time college graduation using a national dataset of 41,359 college applications. Our features include socio-demographics, institutional graduation rates, academic achievement, standardized test scores, engagement in extracurricular activities, and work experiences. We identify five latent classes based on available sociodemographic data and train Random Forest classifiers to successfully predict 4-year graduation. When individually trained and tested on each class using a split-half validation method, we achieved AUROCs between 0.629 and 0.694. We then evaluate how a model trained on the entire dataset performs on each latent class by performing a slicing analysis, finding a 6 to 10 percent improvement in AUROCs compared to the individual-class models. We explore fairness of our model by extending the slicing analysis to consider Absolute Between ROC Area (ABROCA), finding similar values for each of our latent classes. We contemplate how our results might be used to avoid perpetuating biases inherent in college application data.

Keywords

college success, college applications, generalizability, fairness, slicing analysis, National Student Clearinghouse, Common App

1. INTRODUCTION

In 2016, the Obama administration issued a report urging data scientists to explore “how technologies can deliberately or inadvertently perpetuate, exacerbate, or mask discrimination.” [6]. To this point, in recent years, machine learning has come to influence a range of real-world activities, such as detecting credit fraud, financial investing, advertising, and, of course, education.

Stephen Hutt, Margo Gardner, Angela L. Duckworth and Sidney D'Mello "Evaluating Fairness and Generalizability in Models Predicting On-Time Graduation from College Applications" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 79 - 88

These methods make complex, sometimes life changing, decisions based upon training data, often without considering if the training data is biased. This is a critical omission because training data takes advantage of past events, which may be unfairly biased against certain subpopulations, such as those of a particular race, gender, or sexual orientation. Given that the past data may be biased, machine learnt models further perpetuate or even exacerbate these biases. The resultant models can be described as ‘unfair’ because they treat different subpopulations differently. For example, Amazon recently had to decommission their recruitment AI as it favored male applicants for technical jobs [15], ostensibly because in the ten years of hiring data used to train the model, the company had predominantly hired more men than women in technical roles. This biased training data caused the model to negatively evaluate resumes that alluded to the applicant being female (e.g. containing the phrase “women’s chess club captain” or having attended a women’s college).

In the context of education, there has been considerable interest in predicting a range of educational outcomes, such as affect, learning style, likelihood of dropping out of a course, and whether a student will succeed on an upcoming test [18, 46]. In this paper, we examine generalizability and fairness in the specific case of predicting college success. We use the CommonApp-NSC dataset, a 6-year longitudinal de-identified dataset of college application data and graduation outcomes. We first identify five subgroups of applicants based on sociodemographics and then train Random Forest classifiers to predict on-time college graduation from application data. We explore how the models generalize across groups and how fair the models are to each of the groups.

1.1 Background

A college degree offers a wide variety of personal, academic, and economic benefits [3]. For example, 2015 median earnings for young U.S. adults (25-34 years of age) with a Bachelor’s degree were 64% higher than those who had only completed high school, a consistent pattern over the past 15 years [39]. In addition to economic gains for the student, college completion also correlates with economic gains for the nation as a whole [11].

However, based on the latest data, only 40% of first-time, full-time U.S. students graduated with a Bachelor’s degree within four years [41, 52] (60% graduated within six years). Moreover, the academic

achievement gap separating students by race/ethnicity in K12 persists in college. Only 21% of Black and 30% of Hispanic students graduated within 4 years, compared to 44% White and 48% Asian/Pacific Islander students [52].

These numbers suggest that there is considerable room for improvement overall and especially for closing the achievement gap. More to the point of the present focus, they introduce a substantial potential for bias in any machine learning model which uses demographics to predict 4-year college graduation outcomes since these models might inherently predict lower graduation for an underrepresented minority based on historic rates irrespective of their abilities.

In addition to demographics, socioeconomic (SES) factors have been reliably linked to college success [16, 17, 19, 63]. An early 1964 study showed that SES factors such as family income, parental occupation, and parental education levels had a significant effect on college retention and graduation [22]. Other work has reported similar links between SES and college success [1, 34, 60], particularly with respect to the relationship between SES and ethnicity [48]. This signals another potential for bias. Indeed, recent work in machine learning (and even in the popular press) has called for an examination of how SES is used in models [14, 67], especially when making crucial decisions about a person's future.

Beyond SES, academic achievement such as high school GPA and standardized test scores have also been shown to be predictive of college success. In large scale studies [53, 66], including a landmark study with 150,000 students, both standardized test scores and high school GPA predicted college success. However, both of these measures has also been (negatively) linked to SES [54, 68], suggesting potential bias in a more indirect way.

Beyond sociodemographics and cognitive ability, Goodwin and Hein [28] hypothesize that the "X-Factors" such as a can-do attitude, self-discipline, and good study habits are also important for college success. This view aligns with Duckworth [16], Dweck [19], Walton [63], and others who argue that non-cognitive factors such as grit, self-control, mindset, and social belonging [16, 17, 19, 63] are critically important for college success after accounting for sociodemographics and cognitive ability.

One complicating factor is that these non-cognitive traits are difficult to accurately measure. Therefore, admissions counselors must rely on self-reports or informer reports (such as from teachers), which have a number of known biases (see [17] for a review). To address this, there is an interest in more objective measurement approaches. One relevant proxy measure of non-cognitive traits is sustained engagement in extracurriculars during high-school. The rationale for the predictive value of extracurriculars is that they provide a context for the development and demonstration of key non-cognitive characteristics (e.g., initiative [37], identity [21, 37], competence, confidence, and character [9]) linked to academic success. However, extracurriculars might also be inherently biased, in that SES influences the amount and types of available extracurricular opportunities [31, 38]. There is some evidence that work experiences might provide similar benefits as extracurriculars provided youth do not work too much (see [65]). However, low SES students might be the ones more likely to work [61, 64], suggesting that work experiences might also be a biased proxy measure.

In previous work with the CommonApp-NSC data set [33], we trained models that could successfully predict four year graduation using sociodemographics, cognitive ability, and non-cognitive factors. However, we did not consider how our models generalized

between subpopulations or if a population was being treated unfairly. We address this issue here by exploring how the models perform across different sociodemographic groups and evaluate the fairness of our classification methods.

1.2 Related Work

Because the field of generalization is vast [59], we focus specifically on generalization across sociodemographic groups in the context of education. We then go on to discuss fairness for a model – that is whether predicted outcomes for a particular group are consistently negative. Available techniques fall into two groups: (1) methods for evaluating if an existing model is fair; and (2) methods for developing a fair model. Both approaches are discussed here.

In machine learning, cross-validation is performed to improve the likelihood of a model generalizing to new instances, or in educational data mining, to new students (i.e., students not in the training set). However, what about generalizability beyond the student to groups of students? The results might not be so promising. For example, a review by Blanchard [7] indicated that work in intelligent tutoring systems (ITS) and artificial intelligence in education (AIED) overwhelmingly samples from White, educated, industrialized, rich, and democratic (the so-called WEIRD) countries. Blanchard notes that cognitive factors differ greatly across cultural contexts and stresses the importance of expanding research in ITS and AIED to non-WEIRD countries.

Baker and Gowda [4] showed that generalizability is not even guaranteed within communities in the United States of America. Using data from a diverse group of students interacting with an ITS, they found that behaviors that were predictive of disengagement significantly differed between urban, suburban, and rural students. This was further supported in [42], where models trained on urban or suburban students generalized to each other but not to rural students. In contrast, Samei et al, [49] reported that their models of effective classroom discourse generalized across urban and non-urban classrooms. Bosch et al [8] also have had success with generalizability. In work on detecting affect while students interacted with an educational physics game, they found that video-based affect detection generalized across ethnicity (as perceived by human observers) and gender.

Each of the aforementioned approaches explore generalizability by training models on one group and testing on another, a simple yet effective method of validation. However, this approach does not apply when it is infeasible to build models for each sub population, for example, when training data is limited. In this case, we must instead consider how a model trained on all the data performs to individual subgroups of interest. In slicing analysis [50], predictive models are trained on the entire data set and evaluated by "slicing" along subpopulations of interest (such as race, or ethnicity). This allows a researcher to explore if a model is only successful for a certain group (e.g. if a model trained on all data is only accurate for white students).

Gardner et al. [26] recently presented a metric to evaluate fairness within slices. They propose Absolute-Between-ROC Area (ABROCA) for quantifying how a predictive model's performance varies across different student subgroups. This metric evaluates whether a model privileges (provides more accurate classification) or disparately impacts (provides less accurate classification) a subgroup by comparing the group's ROC curve to the ROC curve of a baseline group. In a study analyzing MOOC dropout rate, they show a significant difference in privilege given to males versus

females in machine learnt models across a variety of feature sets and across a classification techniques.

Another method for evaluating models is Individual Fairness [20], which states that in order for a model to be considered fair it must yield similar predictions to similar individuals. The success of this evaluation method depends upon how similarity is defined, a challenging task when the number of predictors is large as in any complex prediction problem.

An alternative to evaluating fairness post-hoc is to design fair models from the ground up. *Fairness through unawareness* posits that a model is fair if it does not include any protected (potentially biased) variables (e.g. [25, 36]). However, this approach ignores the fact that protected variables such as ethnicity may be encoded (via correlation or similar) with variables not initially considered to be protected, such as participation in extracurricular activity or standardized test scores. [27, 32].

Alternatively, Kusner et al. [36] have introduced the idea of Counterfactual Fairness, which requires an understanding of causality among predictors (see [36, 45]). Whereas this method has been successful in datasets with a limited number of variables, understanding causality in a complex dataset presents many challenges and may render this approach unfeasible.

1.3 Contributions of Current Study

The present study is novel in multiple respects. We build upon work predicting on-time Bachelor's graduation solely from information contained in college applications. We derive 143 variables from each application and train models on 41,359 instances. Our sample includes students from all 50 U.S. states as well as international students, yielding many options for exploring generalizability and fairness.

We first cluster the sociodemographic data by identifying latent classes of students within the dataset. From available sociodemographic variables (e.g., race/ethnicity, parent education, parents' marital status, and English language learner status), we identify five distinct latent classes of applicants for further investigation. Specifically, we examine the accuracy of models trained and tested on the same class. We then use a slicing analysis to investigate how a model trained on all the data performs for each of the classes.

It should be noted that our complex, real-world dataset does not easily lend itself to current methods for designing fair models. Decades of research have shown that SES is a predictor of college success (as cited above), so a fairness through unawareness approach would require ignoring an important predictor. Likewise, counterfactual fairness requires understanding causality in the dataset, a challenge given that we are working with 143 variables. We instead evaluate the fairness of models trained with traditional machine learning approaches using ABROCA as the pertinent metric.

2. DATASET

2.1 CommonApp-NSC Data¹

The Common App [55] is a nonprofit organization that hosts a portal where high school students can complete and submit applications to nearly 700 colleges. The Common App streamlines the admissions processes by enabling students to complete one "common" application that can be submitted to multiple colleges

across the country. Whilst individual colleges may have their own supplemental applications (e.g. additional essays), the core application remains the same.

The Common App has three parts. The student section includes information on sociodemographics, future college plans, family history, academic history, standardized test scores, honors received (for academic, sporting, or other pursuits), extracurricular activities, work history, and disciplinary history. Students also submit a personal essay, but these are not available to us due to privacy concerns. A separate evaluation consists of teacher ratings of the student across several dimensions, ranging from "quality of writing" to "reaction to setbacks". Finally, the secondary school report contains information on the student's high school (e.g., percent of graduation class enrolling in college), the student's academic performance (e.g., class rank, GPA), and evaluations from the student's guidance counselor (e.g., ratings of academic achievements, difficulty of courses, and personal qualities).

The National Student Clearinghouse (NSC) is a nonprofit organization created in connection with the financial aid lending industry that gathers enrollment data for student borrowers. The NSC data tracks the following information for each student on a per-semester basis: college name, college type (2/4 year; private or public), enrollment (none, full, part-time), major, and graduation status (degree, and major).

Both organizations have merged, de-identified, and shared the data with us, which we prepared for statistical analyses. The Common App contains individual applications from 413,675 students who completed the 2008 application for admission in the 2009 school year. We successfully matched 362,205 of these applications to 2015 NSC records. From this subset, we removed 50,894 students who enrolled in college prior to 2008 and an additional 3 students due to data integrity issues, leaving 311,308 students.

To account for institutional effects on the probability of graduation, we obtained 4- and 6-year graduation rates from the National Center for Educational Statistics (NCES) [47] for the institution students first enrolled in (i.e., their first entry in the NSC). We used 2012 graduation rates to avoid including students from our 2009 student cohort who would be on-track to graduate with a 4-year degree in 2013. We obtained institution graduation data for 89% of the students, resulting in a reduced sample of 278,201 students.

We also obtained information on students' high school environments (e.g. demographics of the school) from the NCES data [47], using the 2007-2008 school year to avoid direct overlap with our student cohort.

Our data only included applications/reports that were completed online, as there was a paper option in 2008. Of the 278,201 students, only 41,359 had a corresponding teacher evaluation and secondary school report, which contained critical GPA scores as entered by guidance counselors. We presume that a majority of the missing cases were submitted on paper; they were therefore not available to us. Previous work investigated the importance of GPA in predicting college success [33] and found that it did not significantly boost prediction after accounting for the other features. Here, we with this subset for consistency, but do not consider GPA.

¹ In what follows, we provide an abridged description of the dataset, which was originally published in [33].

2.2 Encoding the Application

We extracted 143 features from the application, including auxiliary sources (e.g., NCES data), which we grouped these into the following categories:

Personal and family, 48 features. Features in this category focus primarily on sociodemographics (e.g., ethnicity, sex, number of parents who went to college, etc.).

Academics and standardized tests, 38 features. This category encodes information from the ‘academics’ (e.g., did a student intend to graduate from high-school on time) and ‘standardized tests’ (e.g., SAT scores) section of the student application along with data about the high school environment from the NCES (e.g., teacher-student ratio).

Activities and work experience, 45 features. Students enter information for up to seven extracurricular activities, including the type of the activity, the time commitment, and the school years in which they participated in the activity. In addition, students can enter up to three work experiences, from which we derived features such as number of jobs and hours per week at each job.

Honors, 10 features. Students describe academic and sporting honors received during their high school career. For each honor, we encode the type of honor, the level of the honor (school, state, national or international), and the grade when it was received.

Institutional graduation rates, 2 features. These are the 4- and 6-year graduation rate of the colleges in which students first enrolled.

Our sample of 41,359 students represented all 50 states and included some international students. The students represented 5,678 secondary schools and were enrolled in 1,238 post-secondary institutions. Forty-four percent (44%) graduated within four years of enrollment; this rate aligns with national norms [41].

2.3 Student demographics

Our sample was majority female (56%). Student age was unavailable due to data de-identification, which eliminated birth dates. With regard to ethnicity, 54% of students identified as Caucasian, 8% as African American, 8% as Hispanic, 8% as Asian American, 5% as Asian Indian, 4% as Mexican American, 1% as Native American/Alaskan, and 5% as other ethnicities (students could select multiple ethnicities as well as decline to answer). In terms of home life and education, 96% had two living parents, 77% of students reported living with both parents, 68% had two parents who attended college, and 16% had one parent who attended college. For secondary education, 68% of students reported attending a public high school, 14% a religious high school, 16% an independent high school, 2% a charter school, and 1% were home schooled. The subset of 41,359 students was representative of the full sample of 311,308 students, differing only with respect to the number of parents who attended college [33].

3. LATENT CLASS ANALYSIS

We used latent class analysis (LCA) to identify five clusters of students based on individual sociodemographic characteristics (race/ethnicity, parent education, parents’ marital status, and English language learner status), the race/ethnic composition of students’ high schools (% African American, % Latino, % White, and % Asian American), and whether the school was Title I eligible (a school is eligible if it has high concentration of low income students [23]). We selected these characteristics because they not only paint a relatively comprehensive portrait of socioeconomic status, but also have demonstrated associations with college success (see Introduction). Specifically, White and Asian American

students, students of college educated parents, students with married parents, and students who speak English as a first language have higher on-time graduation rates than are African American and Latino students [58], first generation college students [57], English language learners [35], and students with single parents [44, 51]. Likewise, high schools with large percentages of low-income and minority students, when compared to predominantly White higher-income high schools, often have lower rates of college matriculation and completion [29]. Although often used in generalizability studies (e.g., [8, 24]), we did not include gender in these models as it does not relate to SES and ethnicity (see Discussion).

We used the entire sample size, in this case, all students who attended a public high school ($N=216,133$) for the latent class analysis in order to obtain the most representative clusters. We used complex mixture models with a maximum likelihood estimator in MPlus 7 [40] to identify our latent class structure. Standard errors were adjusted to account for the clustering of students within high schools. An initial two-class solution yielded AIC and BIC values of 385,195.512 and 385,493.737, respectively. Subsequently, we tested solutions with up to six classes. Although each increase in the number of classes resulted in notable improvements in model fit (see Table 1), the magnitude of these improvements diminished with increasing model complexity. The selection of the final five-class solution (see Table 2) balanced model fit against pragmatism. Specifically, the six-class solution fit the data somewhat better than the five-class solution, but two of the six classes had very similar profiles (i.e., profiles similar to class 3 in Table 2). Each class in the five-class solution, on the other hand, had a distinct profile as described below.

Table 1. Model fit by number of latent classes

#	AIC	Incremental reduction in AIC	BIC	Incremental reduction in BIC
2	385195.51	--	385493.74	--
3	196766.61	188428.90	197198.52	188295.21
4	16127.75	180638.87	16693.35	180505.18
5	-103474.54	119602.29	-102775.26	119468.60
6	-180815.24	77340.70	-179982.27	77207.01

Of the 41,359 students analyzed here, only 28,122 were included in the LCA analysis since we only focused on those who attended a public high school. Class 1 contains a plurality of Black students with a sizable white minority (20%) in their high schools. The majority of students are native English speakers, approximately half of students are first generation college students and approximately half of students are children of unmarried parents. This reflects an average SES. Class 2 contains a plurality of White students, but other groups are represented in their high schools (51% white). The majority are native English speakers with married, college-educated parents. Students in this class typically attend a non-Title I eligible, diverse high school where approximately half the students are white with moderate representation across other ethnic/race groups. Thus, this class can be categorized as predominantly white students, high SES students in diverse high schools. Class 3 is similar to Class 2, except with a higher majority of white students and students typically attending a primarily white high school. These students are also high-SES.

Table 2. Five Class Solution: profiles across classification variables by latent class N= 28,122

Latent class/ label	N	Proportion of class that is...										Average high school race/ethnic proportions			
		Proportion of sample					First generation			Attending Title I eligible high school					
		White	Black	Latino	Asian	Other race/ethnicity	college student	married parents	Child of English Language Learner	high school	high school	White	Black	Latino	Asian
1 (Black, mid-SES)	1,745	0.06	0.26	0.10	0.08	0.10	0.52	0.54	0.14	0.40	0.40	0.20	0.61	0.12	0.03
2 (White/diverse, high SES)	5,670	0.20	0.47	0.10	0.13	0.15	0.33	0.75	0.18	0.25	0.25	0.51	0.15	0.18	0.11
3 (White, high SES)	16,959	0.60	0.69	0.02	0.04	0.07	0.23	0.79	0.05	0.20	0.20	0.85	0.04	0.04	0.03
4 (Asian, high SES)	1,051	0.04	0.17	0.02	0.05	0.67	0.10	0.87	0.52	0.20	0.20	0.28	0.05	0.13	0.52
5 (Latino, low SES)	2,697	0.10	0.17	0.11	0.54	0.13	0.70	0.62	0.45	0.77	0.77	0.13	0.12	0.65	0.06

*Note. Proportions of students of varying race/ethnic groups in each class do not sum to 1 due to missing data. N is less than the full sample as we only included students from public high schools

Class 3 is the largest of the classes representing 60% of the students. Class 4 is the smallest of the classes, containing 4% of students, the majority of students are Asian, and a sizable number (52%) of students are English language learners. The majority of students in class 4 have college educated; married parents and attend a non-Title I eligible high school, which suggests high-SES. Finally, in class 5, there is a plurality of Latino students, many (45%) of whom are English language learners. The majority of students have married parents who did not attend college. Most students in this class attended majority Latino, Title I eligible high schools, suggesting low-SES.

4. MACHINE LEARNING

We used the scikit-learn library [43] for machine learning. We focused on Random Forests because previous work that considered logistic regression, naive Bayes, decision tree (using the scikit-learn CART-like algorithm), and gradient-boosted decision trees [33]. found that Random Forest was consistently the best performing approach.

Hyperparameters for the random forest classifier [10, 30], were tuned on the training set using the cross-validated grid search method provided by scikit-learn [43]. Specifically, the number of trees in the forest (`n_estimators`), the maximum number of features to consider when searching for the best split (`max_features`), and the maximum depth of the trees (`max_depth`) were tuned. By careful tuning of these hyperparameters, we negate the need for traditional feature selection, as this is then implicit in the Random Forest algorithm when hyperparameters are set to appropriate values. The random seed was set to a random integer generated by the Numpy.random library [62]. Other hyperparameters relating to limiting the size of the trees (other than maximum depth) were left at default values as resources were sufficient to compute unpruned trees in reasonable time.

We validated our models using a student-level k-fold cross-validation ($k=2$). For each iteration of the classifier, a random 50% of students were assigned to the training set, the remaining 50% to the test set, the process was repeated with the sets reversed, and results computed after pooling predictions across the folds. By using a low k value, we increase the size of our test set, increasing the likelihood that successful models will generalize to new data. This process was repeated for 15 iterations and the results were averaged across iterations. We selected 15 iterations to balance computation time and reliability across multiple training/testing pairs. Although setting $k=2$ imposes a stringent test of the model by removing half the data for the test set, it helps to ensure that the models will generalize to new students.

We note that for some of the latent classes there is a substantial data skew (more instances of not graduating than graduating). Class imbalance poses a challenge because supervised learning methods tend to bias predictions towards the majority class. To compensate for this concern, we used the SMOTE algorithm [12] to create synthetic instances of the minority class by interpolating feature values between an instance and its randomly chosen nearest neighbors until the classes were equated. SMOTE was only applied on the training sets; the original class distributions were maintained in the test sets in order to ensure validity of the results.

5. RESULTS

We report area under the receiver operating characteristic curve (AUROC). Whereas overall recognition rate/model accuracy is susceptible to data skew, AUROC presents the result relative to chance (0.5).

5.1 Generalization

We first compared how models trained on each class individually compared to a model trained upon all data. The all data model was evaluated using a slicing analysis, where predictive model performance is evaluated by “slicing” along subpopulations, in this case, the computed LCA classes. We trained a random forest classifier on all of the data (41,359 instances) and then evaluated it by the five latent classes previously identified. The results of this analysis are shown in Figure 1 with the baseline reflecting chance performance (AUROC of 0.5).

Each of the individual models performed above chance, suggesting that our methods generalized across classes. However, there was a disparity between the classes, the highest performing (class 3, white high SES) performed 10% better than the lowest performing (class 4, Asian high SES). The two lowest AUROC scores were for class 1 (Black, Mid SES) and class 4 (Asian, high SES); the two classes with the lowest number of instances (1,745 and 1,051 respectively).

Our model trained on all students performed better across all classes than individually trained models, with improvements ranging from 6-10%. The difference between the worst performing and best performing groups also decreased to 6%, implying better generalization across groups. One reason for this may be the improved power that comes with a higher number of instances; the all data model was trained on 41,369, instances, more than double that of the largest LCA class (16,959 instances).

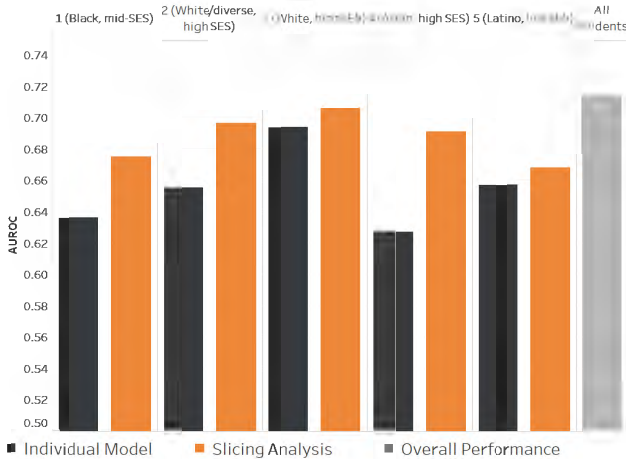


Figure 1. Accuracy of individual models compared to a slicing analysis of a classifier trained on all data

5.2 Fairness

We next examined the fairness of our model. Using the model trained on all data (41,359 instances) we computed five ROC curves, one for each of the latent classes. Recall that this was done for 15 iterations with $k=2$ cross validation. The ROC curves for a single iteration is shown in Figure 2, this iteration was chosen as the ABROCA scores are similar to the averages shown in Table 3. A sixth ROC curve for all students is also shown for comparison.

In order to formally compare two curves, we use Absolute Between ROC Area (ABROCA) [26], defined as:

$$\int_0^1 |ROC_b(t) - ROC_c(t)| dt$$

Here, ROC_b is the baseline curve and ROC_c is the comparison curve. ROC curves characterize model accuracy as the likelihood of correct positive predictions versus the likelihood of false positive predictions. ABROCA measures the absolute difference between two curves, allowing for the possibility that the curves may cross each other (see [26] for details). A higher ABROCA value between two groups implies a higher difference in predictions and thus more unfairness in the model.

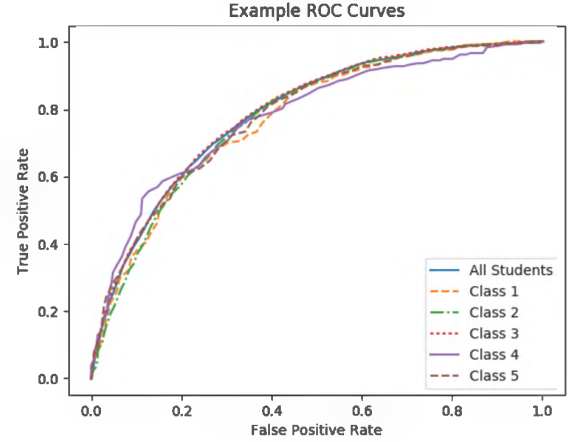


Figure 2. Sample ROC curves for each Latent class from model trained on 41,359 instances

We used class 3 (White, high SES) as the baseline ROC curve as it is the highest performing group in our dataset. It is also a group typically overrepresented in Educational Data Mining [7]. When compared to this class, the other classes had low ABROCA values (see Table 3), perhaps unsurprising given the similarity of the curves in Figure 2. In general, the ABROCA values were all low with only small differences between classes, leading us to conclude that our model was providing fair predictions across our sociodemographic groups.

Table 3. Slicing analysis by latent class from model trained on all data

LCA Class	AUROC	ABROCA
1 (Black, mid-SES)	0.675	0.011
2 (White/diverse, high SES)	0.696	0.005
3 (White, high SES)	0.706	-
4 (Asian, high SES)	0.691	0.016
5 (Latino, low SES)	0.668	0.008

6. DISCUSSION

On-time 4-year college graduation is something of a “holy-grail” for students, parents, and educators alike [58]. Although efforts to improve college enrollment have been paying off, graduation rates are still lackluster with troublesome achievement gaps stubbornly persisting. Big data approaches might offer a potential solution to improving college graduation rates by providing new insights into the “ingredients” of success. However, they have their own set of limitations and biases, which need to be addressed before we uncover their full potential. Accordingly, we investigated how

predictive models of 4-year college graduation generalized across sociodemographic subgroups identified through latent class analysis and whether they yielded fair predictions for the different groups of students.

6.1 Main Findings

Whilst much of the previous work has relied on limited datasets and traditional statistical techniques [53, 61, 69], we harness a large and diverse dataset with greater potential for generalizability. Specifically, using data from students' college application, we have been able to predict college graduation with moderate accuracy across demographic subgroups. We also show through slicing analysis that a model trained on all data generalizes across all of the subgroups and outperforms individual models (improvement ranging from 6-10%). Training a model on all of the data also reduced the disparity between the subgroups.

By evaluating the ABROCA metric, we were also able to examine which of our subgroups (if any) the classifier was treating unfairly. An unfair model would perform generate less accurate predictions for a given subgroup compared to the baseline group (White High SES in our case). In general, the differences in ABROCA scores were small, suggesting that our model treats no one class significantly different from another.

Whilst all of our models' predictions were substantially more accurate than a chance, there were still inaccuracies. In many ways, this result is reassuring, as we have only considered data from high school. The error that exists across all of our models confirm that college success does not merely depend on a student's environment, past achievement, and experiences. What students experience and do in college plays a critical role in their success. Simply put, there is no predetermination. This gives us hope that through careful data mining we can soon begin to close the achievement gaps that exist across different sociodemographics.

6.2 Applications

It is perhaps easiest to start with how these models should *not* be used. Specifically, the models should not be used to make college admissions decisions because their accuracy scores are insufficient to drive life-changing decisions for individual students and they do not capture several additional factors of the college years that are important for success (e.g. financial needs, life-altering events, social pressures).

We show that it is possible to build generalizing and fair detectors in this domain. On a larger scale, we hope to use this research to provide actionable advice for educators so that they may better prepare students for college success. Further analysis is needed to derive these personalized recommendations, especially since the current models are correlational and thereby unsuited for causal inference.

There are further applications at the college level. Many U.S. colleges have committed to improving 4-year graduation rates [13]. This has resulted in an increased reliance on educational data mining approaches, especially methods to identify "at-risk" students early on [2]. A common issue however is that early warnings are not early enough [5]. Our models consider college application data, so enable us to pre-identify students who might need additional support before they begin their studies. Of course, the models' assessments should be privately communicated to the student's themselves and perhaps to a trusted counselor so they are empowered to take whatever next step is in their best interests.

6.3 Limitations and Future Work

All studies have limitations and ours is no exception. Each of the latent classes had different graduation rates and varied number of instances (a difference of 15,909 instances between the smallest and the largest groups). We attempted to account for class imbalance via synthetic oversampling. However, further work is required to evaluate how the number of instances influenced our results. Future work will also explore the effect of increasing the amount of data used to train models.

Second, our sample only included students who applied to schools that accepted the Common App, which would introduce selection bias, which we cannot account for in this work. Further study is required to investigate how the results generalize to other colleges in the U.S. and beyond.

In addition to addressing these limitations, there are also several promising avenues for future work. For example, since biased variables seem to be predictive in this domain, we will also look into ways to create fair models without fully ignoring the biased variables, perhaps by deriving unbiased proxies.

Our models utilized a range of features including socioeconomic factors, academic history, cognitive ability, the high school environment, and indicators of extracurricular participation that may reflect non-cognitive characteristics. Previous work using the CommonApp-NSC dataset work has shown that different feature groups [33] achieve different classification accuracies. In future work we intend to explore fairness for different feature groups and incorporate insights into the design of fairer models.

When computing the LCA classes, we did not include gender as a variable. However, gender might be more relevant when it comes to specialized outcomes, such as STEM graduation where there are significant disparities across the genders. Relatedly, we also will explore other outcome metrics such as 6-year graduation and STEM graduation, an area with wide achievement gaps when it comes underrepresented groups [56].

6.4 Concluding Remarks

In conclusion, the age of big data brings with it big opportunity, and big responsibility. Although a predictive modeling approach applied to big data has considerable potential in providing new insights to illuminate persistent challenges, these methods have own weaknesses, particularly when it comes to making biased predictions. Thus, we must also consider how our models are perpetuating pre-existing bias and how this can be prevented. Taking the case of predicting on-time college graduation outcomes, we show that our models both generalize and are fair to various sociodemographics subgroups, a critical step towards using these models more broadly.

ACKNOWLEDGMENTS

We are grateful to the Common App for providing us with the dataset, which made this work possible. We also thank Parker Goyer for her assistance in coding the NSC data and to Tammer Ibrahim for his help with the computational infrastructure. This research was supported by the Walton Family Foundation, the Mindset Scholars Network, the Bill & Melinda Gates Foundation, the Joyce Foundation, the Overdeck Family Foundation, and the Raikes Foundation. Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

- [1] Allen, D. 1999. Desire to finish college : an empirical link between motivation and persistence. *Research in Higher Education*. 40, 4 (1999), 461–485. DOI:<https://doi.org/10.1023/A:1018740226006>.
- [2] Allensworth, E. and Easton, J.Q. 2007. What matters for staying on-track and graduating in chicago public high schools: a close look at course grades, failures, and attendance in the freshman year. *Consortium on Chicago School Research*. (2007), 1–61.
- [3] Astin, A.W. 1977. *Four critical years. effects of college on beliefs, attitudes, and knowledge*. Jossey-Bass Publishers.
- [4] Baker, R.S.J. d. and Gowda, S.M. 2010. An analysis of the differences in the frequency of students' disengagement in urban, rural, and suburban high schools. *Proceedings of the 3rd International Conference on Educational Data Mining*. (2010), 11–20.
- [5] Beaudoin, B. and Kumar, P. 2012. Using data to identify at-risk students and develop retention strategies. *EAB Custom Research Brief*. (2012).
- [6] Big Risks, Big Opportunities: the Intersection of Big Data and Civil Rights | whitehouse.gov: 2016. <https://obamawhitehouse.archives.gov/blog/2016/05/04/big-risks-big-opportunities-intersection-big-data-and-civil-rights>. Accessed: 2019-02-22.
- [7] Blanchard, E.G. 2012. On the weird nature of its/aied conferences: a 10 year longitudinal study analyzing potential cultural biases. *Intelligent Tutoring Systems* (2012), 280–285. DOI:https://doi.org/10.1007/978-3-642-30950-2_36.
- [8] Bosch, N. et al. 2016. Using video to automatically detect learner affect in computer-enabled classrooms. *ACM Transactions on Interactive Intelligent Systems* (2016), 1–26. DOI:<https://doi.org/10.1145/2946837>.
- [9] Bowers, E.P. et al. 2010. The five cs model of positive youth development: a longitudinal analysis of confirmatory factor structure and measurement invariance. *Journal of Youth and Adolescence*. 39, 7 (2010), 720–735. DOI:<https://doi.org/10.1007/s10964-010-9530-9>.
- [10] Breiman, L. 2001. Random forests. *Machine Learning*. 45, 1 (2001), 5–32. DOI:<https://doi.org/10.1023/A:1010933404324>.
- [11] Carnevale, A.P. et al. 2013. *The college payoff: education, occupations, lifetime earnings*. Center on Education and the Workforce.
- [12] Chawla, N. V. et al. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*. 16, 1 (Jun. 2002), 321–357. DOI:<https://doi.org/10.1613/jair.953>.
- [13] Complete College America 2014. *Four-year myth*. Complete College America.
- [14] Crawford, K. and Calo, R. 2016. There is a blind spot in ai research. *Nature*. (2016). DOI:<https://doi.org/10.1038/538311a>.
- [15] Dastin, J. 2018. Amazon scraps secret ai recruiting tool that showed bias against women. *Reuters*.
- [16] Duckworth, A.L. and Allred, K.M. 2012. Temperament in the classroom. *Handbook of temperament*. Guilford Press New York, NY. 627–644.
- [17] Duckworth, A.L. and Yeager, D.S. 2015. Measurement matters: assessing personal qualities other than cognitive ability for educational purposes. *Educational Researcher*. 44, 4 (2015), 237–251. DOI:<https://doi.org/10.3102/0013189X15584327>.
- [18] Dutt, A. et al. 2017. A systematic review on educational data mining. *IEEE Access*. 5, (2017), 15991–16005. DOI:<https://doi.org/10.1109/ACCESS.2017.2654247>.
- [19] Dweck, C.S. 2006. *Mindset: the new psychology of success*. Random House.
- [20] Dwork, C. et al. 2012. Fairness through awareness. *Proceedings of the 3rd innovations in theoretical computer science conference* (2012), 214–226. DOI:<https://doi.org/10.1145/2090236.2090255>.
- [21] Eccles, J.S. et al. 2003. Extracurricular activities and adolescent development. *Journal of Social Issues*. 59, 4 (2003), 865–889. DOI:<https://doi.org/10.1086/223736>.
- [22] Eckland, B.K. 1964. Social class and college graduation : some misconceptions corrected. *American Journal of Sociology*. 70, 1 (1964), 36–50. DOI:<https://doi.org/10.1086/223736>.
- [23] Epstein, J.L. and Hollifield, J.H. 2005. Title i and school-family--community partnerships: using research to realize the potential. *Journal of Education for Students Placed at Risk (JESPAR)*. (2005). DOI:https://doi.org/10.1207/s15327671espr0103_6.
- [24] Ewert, S. 2012. Fewer diplomas for men: the influence of college experiences on the gender gap in college graduation. *The Journal of Higher Education*. 83, 6 (2012), 824–850. DOI:<https://doi.org/10.1353/jhe.2012.0042>.
- [25] Gajane, P. and Pechenizkiy, M. 2017. On formalizing fairness in prediction with machine learning. *arXiv preprint arXiv:1710.03184*. (2017).
- [26] Gardner, J. et al. 2019. Evaluating the fairness of predictive student models through slicing analysis. *Proceedings of the 10th Conference on Learning Analytics and Knowledge* (Tempe, AZ, USA, 2019), 10. DOI:<https://doi.org/10.1145/3303772.3303791>.
- [27] Gardner, M. et al. 2008. Adolescents' participation in organized activities and developmental success 2 and 8 years after high school: do sponsorship, duration, and intensity matter? *Developmental psychology*. 44, 3 (2008), 814–830. DOI:<https://doi.org/10.1037/0012-1649.44.3.814>.
- [28] Goodwin, B. and Hein, H. 2016. Research says/the x factor in college success. *Educational Leadership*. 73, 6 (2016), 77–78.
- [29] High school demographics continue to impact college success: 2016. <https://studentclearinghouse.org/blog/high-school-demographics-continue-to-impact-college-success/>.
- [30] Ho, T.K. 1995. Random decision forests. *Proceedings of the Third International Conference on Document Analysis and Recognition* (Washington, DC, USA, 1995), 278–282.

- DOI:<https://doi.org/10.1109/ICDAR.1995.598994>.
- [31] Holland, A. and Andre, T. 1987. Participation in extracurricular activities in secondary school: what is known, what needs to be known? *Review of Educational Research*. 57, 4 (1987), 437–466. DOI:<https://doi.org/10.3102/00346543057004437>.
- [32] Humbert, M.L. et al. 2006. Factors that influence physical activity participation among high-and low-ses youth. *Qualitative health research*. 16, 4 (2006), 467–483. DOI:<https://doi.org/10.1177/1049732305286051>.
- [33] Hutt, S. et al. 2018. Prospectively predicting 4-year college graduation from student applications. *Proceedings of the 8th International Conference on Learning Analytics and Knowledge* (New York, NY, USA, NY, USA, 2018), 280–289. DOI:<https://doi.org/10.1145/3170358.3170395>.
- [34] Ishitani, T.T. 2006. Studying attrition and degree completion behavior among first-generation college students in the united states. *The Journal of Higher Education*. 77, 5 (2006), 861–885. DOI:<https://doi.org/10.1353/jhe.2006.0042>.
- [35] Kanno, Y. and Cromley, J.G. 2013. English language learners’ access to and attainment in postsecondary education. *TESOL Quarterly*. (2013). DOI:<https://doi.org/10.1002/tesq.49>.
- [36] Kusner, M.J. et al. 2017. Counterfactual fairness. *Advances in Neural Information Processing Systems* 30 (2017), 4066–4076.
- [37] Larson, R.W. et al. 2006. Differing profiles of developmental experiences across types of organized youth activities. *Developmental psychology*. 42, 5 (2006), 849–863. DOI:<https://doi.org/10.1037/0012-1649.42.5.849>.
- [38] Marsh, H. and Kleitman, S. 2002. Extracurricular school activities: the good, the bad, and the nonlinear. *Harvard Educational Review*. 72, 4 (2002), 464–515. DOI:<https://doi.org/10.17763/haer.72.4.051388703v7v7736>.
- [39] McFarland, J. et al. 2017. The condition of education 2017. *National Center for Education Statistics*. (2017).
- [40] Muthén, L.K. and Muthén, B.O. 2015. *Mplus. seventh edition*.
- [41] NCES 2016. Digest of education statistics 2016. table 326.10. graduation rate from first institution attended for first-time, full-time bachelor’s degree- seeking students at 4-year postsecondary institutions. *Digest Of Education Statistics*. U.S. Department of Education, National Center for Education Statistics.
- [42] Ocumpaugh, J. et al. 2014. Population validity for educational data mining models: a case study in affect detection. *British Journal of Educational Technology*. (2014). DOI:<https://doi.org/10.1111/bjet.12156>.
- [43] Pedregosa, F. et al. 2011. Scikit-learn: machine learning in python. *Journal of Machine Learning Research*. 12, (2011), 2825–2830.
- [44] Ver Ploeg, M. 2002. Children from disrupted families as adults: family structure, college attendance and college completion. *Economics of Education Review*. (2002). DOI:[https://doi.org/10.1016/S0272-7757\(00\)00050-9](https://doi.org/10.1016/S0272-7757(00)00050-9).
- [45] Russell, C. et al. 2017. When worlds collide: integrating different counterfactual assumptions in fairness. *Advances in Neural Information Processing Systems* 30 (2017), 6414–6423.
- [46] Ryan S.J.d. Baker, K.Y. 2009. The state of educational data mining in 2009: a review and future visions. *Journal of Educational Data Mining*. (2009). DOI:<https://doi.org/10.1109/ASE.2003.1240314>.
- [47] Sable, J. and Plotts, C. 2010. Documentation to the nces common core of data public elementary/ secondary school universe survey: school year 2007–08 (nces 2010-302rev). (2010).
- [48] Saegert, S.C. et al. 2007. *Report of the APA Task Force on Socioeconomic Status APA Task Force on Socioeconomic Status*.
- [49] Samei, B. et al. 2015. Modeling classroom discourse: do models that predict dialogic instruction properties generalize across populations? *Educational Data Mining* (2015), 444–447.
- [50] Sculley, D. et al. 2018. Winner’s curse? on pace, progress, and empirical rigor. *International Conference on Learning Representations* (2018).
- [51] Sigle-Rushton, W. and McLanahan, S. 2004. Father absence and child well-being: a critical review. *The future of the family*. (2004).
- [52] Snyder, T.D. et al. 2016. *Digest of education statistics, 2015*. National Center for Education Statistics, Institute of Education Sciences, U.S. Department of Education.
- [53] Stumpf, H. and Stanley, J.C. 2002. Group data on high school grade point averages and scores on academic aptitude tests as predictors of institutional graduation rates. *Educational and Psychological Measurement*. 62, 6 (2002), 1042–1052. DOI:<https://doi.org/10.1177/0013164402238091>.
- [54] Tate, W.F. 2006. Race-ethnicity, ses, gender, and language proficiency trends in mathematics achievement: an update. *Journal for Research in Mathematics Education*. (2006). DOI:<https://doi.org/10.2307/749636>.
- [55] The Common Application: <http://www.commonapp.org/>. Accessed: 2017-09-27.
- [56] Thompson, R. and Bolin, G. 2011. Indicators of success in stem majors: a cohort study. *Journal of College Admission*. (2011).
- [57] Tinto, V. 2008. Moving beyond access: college success for low-income, first-generation students. *The Pell Institute for the Study of Opportunity in Higher Education*. (2008).
- [58] US Department of Education, N. 2018. The condition of education - 2018. *Institute of Education Sciences*. (2018).
- [59] Vidyasagar, M. 2002. *A theory of learning and generalization*. Springer-Verlag.
- [60] Walpole, M. 2008. Emerging from the pipeline : african american students , socioeconomic status , and college experiences and outcomes. *Research in Higher Education*. 49, 3 (2008), 237–255. DOI:<https://doi.org/10.1007/s11162-007-9079-y>.
- [61] Walpole, M. 2003. Socioeconomic status and college: how ses affects college experiences and outcomes. *The Review*

- of *Higher Education*. 27, 1 (2003), 45–73. DOI:https://doi.org/10.1353/rhe.2003.0044.
- [62] van der Walt, S. et al. 2011. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*. 13, 2 (Mar. 2011), 22–30. DOI:https://doi.org/10.1109/MCSE.2011.37.
- [63] Walton, G.M. and Cohen, G.L. 2011. A brief social-belonging intervention improves academic and health outcomes of minority students. *Science (New York, N.Y.)*. 331, 6023 (Mar. 2011), 1447–1451. DOI:https://doi.org/10.1126/science.1198364.
- [64] Warren, J.R. et al. 2011. Employment during high school: consequences for students’ grades in academic courses. *Journal of Vocational Education Research*. (2011). DOI:https://doi.org/10.5328/jver26.3.366.
- [65] Warren, J.R. 2002. Reconsidering the relationship between student employment and academic outcomes: a new theory and better data. *Youth & Society*. 33, 3 (2002), 366–393. DOI:https://doi.org/10.1177/0044118X02033003002.
- [66] Waugh, G. et al. 1994. Using ethnicity, sat/act scores, and high school gpa to predict retention and graduation rates. *Florida Association for Institutional Research Conference*. (1994).
- [67] Zou, J. and Schiebinger, L. 2018. AI can be sexist and racist — it’s time to make it fair. *Nature*. (2018). DOI:https://doi.org/10.1038/d41586-018-05707-8.
- [68] Zwick, R. 2002. Is the sat a “wealth test”? *Phi Delta Kappan*. 84, 4 (2002), 307–311. DOI:https://doi.org/10.1177/003172170208400411.
- [69] Zwick, R. and Sklar, J.C. 2005. Predicting college grades and degree completion using high school grades and sat scores: the role of student ethnicity and first language of student ethnicity and first language. *American Educational Research Journal*. 42, 3 (2005), 439–464. DOI:https://doi.org/10.3102/00028312042003439.

Measuring students' thermal comfort and its impact on learning

Han Jiang
Worcester Polytechnic Institute
hjiang@wpi.edu

Matthew Iandoli
Worcester Polytechnic Institute
mjiandoli@wpi.edu

Steven Van Dessel
Worcester Polytechnic Institute
svandessel@wpi.edu

Shichao Liu
Worcester Polytechnic Institute
sliu8@wpi.edu

Jacob Whitehill
Worcester Polytechnic Institute
jrwhitehill@wpi.edu

ABSTRACT

Thermal comfort (TC) – how comfortable or satisfied a person is with the temperature of her/his surroundings – is one of the key factors influencing the *indoor environmental quality* of schools, libraries, and offices. We conducted an experiment to explore how TC can impact students' learning. University students ($n = 25$) were randomly assigned to different temperature conditions in an office environment ($25^{\circ}\text{C} \rightarrow 30^{\circ}\text{C}$, or $30^{\circ}\text{C} \rightarrow 25^{\circ}\text{C}$) that were implemented using a combination of heaters and air conditioners over a 1.25 hour session. The task of the participants was to learn from tutorial videos on three different topics, and a test was given after each tutorial. The results suggest that (1) changing the room temperature by a few degrees Celsius can stat. sig. impact students' self-reported TC; (2) the relationship between TC and learning exhibited an inverted U-curve, i.e., should be neither too uncomfortable nor too comfortable. We also explored different computer vision and sensor-based approaches to measure students' thermal comfort automatically. We found that (3) TC can be predicted automatically either from the room temperature or from an infra-red (IR) camera of the face; however, (4) TC prediction from a normal (visible-light) web camera is highly challenging, and only limited predictive power was found in the facial expression features to predict thermal comfort.

Keywords

thermal comfort, automated face analysis

1. INTRODUCTION

Most of the time that people learn takes place indoors. Primary and secondary school students are typically in school buildings for most of the day and do homework in their houses and apartments in the evenings. Adult learners may learn as part of their job in an office or pursue lifelong-learning opportunities at home. The *indoor environment*

quality (IEQ) of where people learn, study, and work can have a significant impact on their physical well-being as well as their cognitive performance [1, 2].

The impact of IEQ on *learning* in particular has a special importance and has begun to interest architects, civil engineers, and educational psychologists in recent years [13]: Young learners in particular might be more sensitive to the influence of the environment due to their age or other physiological characteristics than adults. Students spend many hours each day in schools; however, since students typically have little control over their schools' physical environment, learners may feel great concern about their thermal comfort [8]. Thermal comfort (TC), which is a key component of IEQ, has been defined as "that condition of mind that expresses satisfaction with the thermal environment and is assessed by subjective evaluation" [3]. Prior work (see section below) has shown that suboptimal thermal comfort conditions can negatively affect students' learning. However, to our knowledge, no study to-date has explored the relationship between the impact of TC on learning and *time*. Is it possible that the effect of suboptimal TC could be mild during brief periods of learning but become more severe as the learning session continues? This is one of the questions we explore in this paper.

Measuring thermal comfort: Different people can experience the same temperature and environment differently, and just because one person has a high degree of thermal comfort does not mean her/his friend or peer will. Since thermal comfort is about a person's *satisfaction* with the thermal comfort, it depends not only on the environment itself, but also on the person's physiological and psychological *adaptability* [9, 7] to her/his environment. How adaptive a person is depends, in turn, on how and where a person grew up, e.g., her/his country of origin and its associated climate.

Due to the partially subjective nature of TC, most studies that sought to measure TC used questionnaires [12, 11, 9, 7]. While these are useful, they suffer from drawbacks such as (1) lack of temporal specificity, (2) recency/primacy effects, (3) disruption to regular activities. These can all lead to inaccurate measurements. Therefore, many researchers have explored alternative approaches based on various sensors (e.g., skin-based temperature sensors, cameras) to measure TC automatically [34, 23, 25, 22, 15, 17].

Han Jiang, Matthew Iandoli, Steven Van Dessel, Shichao Liu and Jacob Whitehill "Measuring students' thermal comfort and its impact on learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 89 - 98

Automatic facial expression recognition: One of the new forms of human observation that has been enabled by advances in machine learning and computer vision is based on automatic facial expression recognition. With technology, it is possible to automatically detect pain in the human body [18], student engagement [36], driver fatigue [33], and many other affective and cognitive states. Inspired by these studies, we explore in this paper whether automatic analysis of facial expression can help to detect a person’s degree of thermal comfort.

Contributions: In our study, we (1) conduct a randomized experiment to explore the relationship between thermal comfort, the time-on-task, and learning. We also (2) explore different sensors and algorithmic approaches to estimating a person’s thermal comfort automatically.

2. RELATED WORK

During the past 10 years there has been substantial interest (see [13, 26] for literature surveys) in measuring the impact of the IEQ on students’ learning. In Table 1 we categorize the prior work on this subject in terms of IEQ factor (light, air, etc.) as well as the method of measuring learning (subjective impression (SI), test (T) performance, school scores (SS), and randomized experiment (RE)). In addition to studies specifically about thermal comfort (TC) [38], other factors of the IEQ such as lighting, air quality, and noise have been considered. Within this research domain, an important dimension of variability is how learning was measured – by asking participants their subjective impressions, from their school scores, or from a test conducted within the experiment itself. Another dimension of variability is whether the study was observational (i.e., compute a correlation between historical data of the IEQ and historical data of learning) or experimental (i.e., randomly assign participants to conditions). The latter is a generally considered to be the more powerful approach since it avoids many potential confounds (e.g., student engagement) and is the approach we pursue in our study.

2.1 Impact of TC on learning

[20, 8] used subjective impression as the learning performance. They both analyzed the relationships between the IEQ (light, air quality, thermal comfort and noise) and learning. [20] found that the learning performance was negatively correlated with the number of student complaints about IEQ. [8] also explored the students’ satisfaction with IEQ, as well as the TC in particular, from survey data gathered from 631 university students. The results showed that satisfaction of IEQ of the classroom was related to the perceived effect of IEQ on learning. [27] conducted a 1-month test during May-June 2012 at a university in Romania. 18 students’ test results of concentrated attention tests (Kraepelin test) and distributive attention test (Prague test)[31, 30] were recorded. The conductors used room temperature, relative humidity and CO₂ concentration to predict test scores. Their results suggested that these indoor environment factors could strongly impact students’ learning performance. [35] conducted an experiment to explore the impact of air temperature on students’ performance. The results indicated that with the same accuracy, students would increase their speed when performing the language-based and numerical performance tasks if the room temperature was re-

duced from 25°C to 20°C in late summer. [24] randomly assigned the participants into different conditions to perform a computer-based reading and learning task. They found that TC had a low and non-significant relationship with the performance; the participants in the extreme condition believed that the temperature had a larger negative impact on their performance than the participants in a normal condition. In [16], the researchers conducted an experiment to explore the impact of TC in 1-on-1 cognitive tasks when students are with a tutor. All the participants experienced all temperature conditions (10°C, 14°C, 15°C, 16°C, 18°C, 20°C). Their experiment indicated that there was an inverted-U relationship between thermal sensation and pupils’ learning performance. A seven point scale of thermal sensation, according to [3], was used. The meaning of the number from -3 to 3 was “cold”, “cool”, “slightly cool”, “neutral”, “slightly warm”, “warm” and “hot” successfully. The results showed that students’ performance was better in the cool or slightly cool conditions compared to the hot condition.

2.2 Measuring thermal comfort

How to measure thermal comfort has been explored for many years. While questionnaires from each person about her/his own TC is useful, they can be inconvenient and tedious. Researchers have thus sought to devise alternative measures that can be measured automatically from various sensors.

Environmental sensors: For instance, the PMV-PPD model, proposed by [12, 11], uses air temperature, mean radiant temperature, air velocity, humidity, and human variables to calculate the Predicted Mean Vote (PMV) of a group of people’s averaged thermal sensation according to [3]. The Predicted Percentage of Dissatisfied (PPD) utilizes PMV to calculate the percentage of people who might complain about their thermal environment.

Body sensors: [34] used skin temperature sensors to collect upper extremity (finger, hand, forearm) skin temperatures and explored how these temperatures related to thermal sensation. [23] explored different configurations of where to place the temperature sensors on the body and identified particular configurations that were most effective.

Cameras: More recently, with the development of machine vision, researchers also explored predicting thermal comfort through cameras. [25] showed that the averaged forehead temperature from infrared (IR) images was correlated with people’s thermal sensation and thermal comfort. [15, 17] leveraged the human thermoregulation process and then applied Eulerian Video Magnification algorithm[37] to filter the visible-light RGB images to predict thermoregulation states, which is one indicator of thermal comfort.

3. EXPERIMENT

In order to assess the impact of thermal comfort on learning and how this effect could change over time, we conducted a laboratory-based learning experiment (approved by WPI’s IRB #18-0372) in which university students ($n = 25$) watched three lecture videos, answered surveys on their thermal comfort, and completed a quiz on what they learned. During the experiment, the indoor environment conditions were monitored and controlled according to a schedule defined by each participant’s randomly assigned experimental

Table 1: Related Work about the impact of indoor environment factors on learning. SI: subjective impression; T: test; SS: school scores; RE: randomized experiment

	Light	Air	Thermal comfort	Noise	Other
SI	Lee, et al.[20] Choi, et al.[8] Marchand, et al.[24]	Kameda, et al.[19] Lee, et al.[20] Choi, et al.[8]	Lee, et al.[20] Choi, et al.[8] Marchand, et al.[24]	Lee, et al.[20] Choi, et al.[8] Marchand, et al.[24]	
T	Dorizas, et al.[10]	Kameda, et al.[19] Dorizas, et al.[10] Sarbu & Cristian.[27]	Dorizas, et al.[10]	Dorizas, et al.[10]	
SS		Haverinen-Shaughnessy, et al.[14]			Barrett, et al.[6] Barrett, et al.[5]
RE	Marchand, et al.[24]	Wargoeki & David.[35]	Wargoeki & David.[35] Marchand, et al.[24] Jiang, et al.[16]	Marchand, et al.[24]	

condition. We also deployed a variety of sensors – camera, environmental, and body – to measure the temperature of the environment and of each participant. These sensor measurements, along with participants’ survey responses, allow us also to explore different automated approaches to estimating a person’s thermal comfort.

3.1 Recruitment of participants

We recruited participants for the experiment through an email list at our university. In the end, 25 students (of whom 9 were female) participated in our experiment. All of them were either undergraduate or graduate students. Each participant was paid for \$20 gift card for his/her participation.

3.2 Procedure

This experiment was conducted on each participant individually and was divided into four sessions. Each session was 21 minutes. Therefore, every participant would sit at a desk around 84 minutes in total. In the first session (adaptation session), each participant gave informed consent, placed the skin-based temperature sensors on her/his body, and listened to the experimenter’s instructions. The purpose of the adaptation session was to neutralize the potential impact of the outside weather conditions or physical activity (e.g., running to class) before the experiment. In each of the remaining three sessions, the participant watched a tutorial video (10 minutes), answered a quiz about it (<5 minutes), completed a thermal comfort survey (<5 minutes), and then took a break. The length of the break (21 min – VideoLength – QuizTime – SurveyTime) depended on how long the participant took to complete the quiz and survey. The order of the tutorial videos was randomized, as was the order of the temperature conditions (warm to neutral, or neutral to warm); see Conditions subsection below. Sensor measurements, including video of the face, were recorded throughout all three tutorial sessions.

After the participant finished putting on the body sensors, the experimenter started the videorecording from the laptop-based web camera, typed the participant’s ID into the webpage, turned the time controller on, and then asked the participant to press the “Start” button whenever she/he was ready. The experimenter then left the room and stayed in the room next-door throughout the rest of the experiment. Using remote access software, the experimenter took an IR image of the participant at the beginning of each tu-



Figure 1: Experimental setup of the desk, laptop, and cameras.

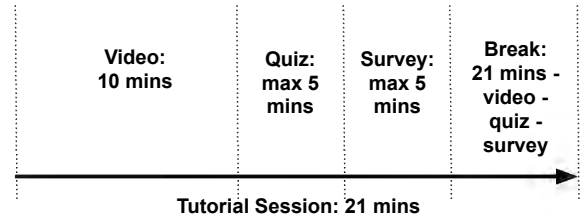


Figure 2: Tutorial session procedure

torial video during the tutorial sessions. See Figure 2 for a schematic of the procedure.

3.3 Environmental controls

We used 4 heaters (to increase the room temperature) and 1 air conditioner (to decrease temperature). In order to maintain the temperature at a constant level, we also deployed 3 thermal controllers. Moreover, in order to change the room temperature (from either warm to neutral, or neutral to warm), we also used 4 timers. To maintain the room temperature to be at least 25°C, 1 heater was always turned on. 3 thermal controllers and 3 timers were connected to the other heaters. The thermal controllers were used to keep the room temperature around 30°C. Timers were used to control when the heaters and air conditioners were turned on and off. The heaters and air conditioner were oriented so that the air did not blow directly onto the participant.

3.4 Sensors

All sensors were adjusted carefully before we started our experiment. They are listed as follows:

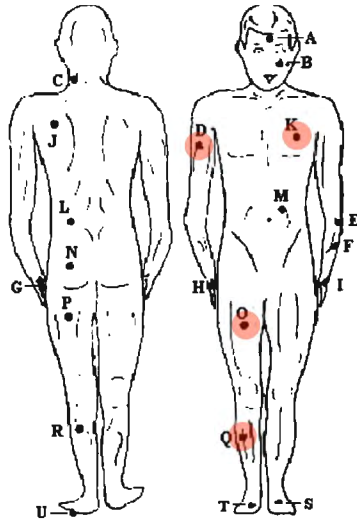


Figure 3: Positions of skin-based temperature sensors on the body.

1. 4 skin temperature sensors. We followed the positions in [23] (see Figure 3). These sensors were used to measure the participant's body temperatures at four different body locations and record the temperature every 1 minute. Sensors were attached using medical tape.
2. Room temperature sensors. These sensors were used to measure the room air temperature at different heights (0.1m, 0.6m, 1.1m and 1.7m) and recorded every 1 minute.
3. 1 web camera on the laptop pointed at the participant's face. Note that the video was lost for 1 out of 25 participants; hence, for our experiments on using the web camera to predict thermal comfort, $n = 24$.
4. 1 infrared (IR) camera pointed at the participant's face. The camera recorded only images, not video. Using the camera's temperature calibration software, the IR images can be used to estimate the participant's face temperature directly.

3.5 Materials

Tutorial videos: We used three 10 min-long tutorial videos and quizzes that were used in a prior study by [32]. The order in which the tutorial videos were presented to each participant was randomized; this was necessary to remove the potential confound that the subject matter, rather than the thermal comfort or time during the learning session, influenced the learning gains. All videos were about social, philosophical, and ethical issues: (1) honesty, (2) language and thought, and (3) empathy.

Thermal comfort survey: We used the same thermal comfort questionnaire survey as in [22, 21]. The survey asks questions such as, "Rate your whole body thermal sensation", "Rate your thermal body comfort", "How sleep/alert do you feel?", and "How easy/difficult is it to concentrate?" The scale was from -3 to +3 with a resolution of 0.1.

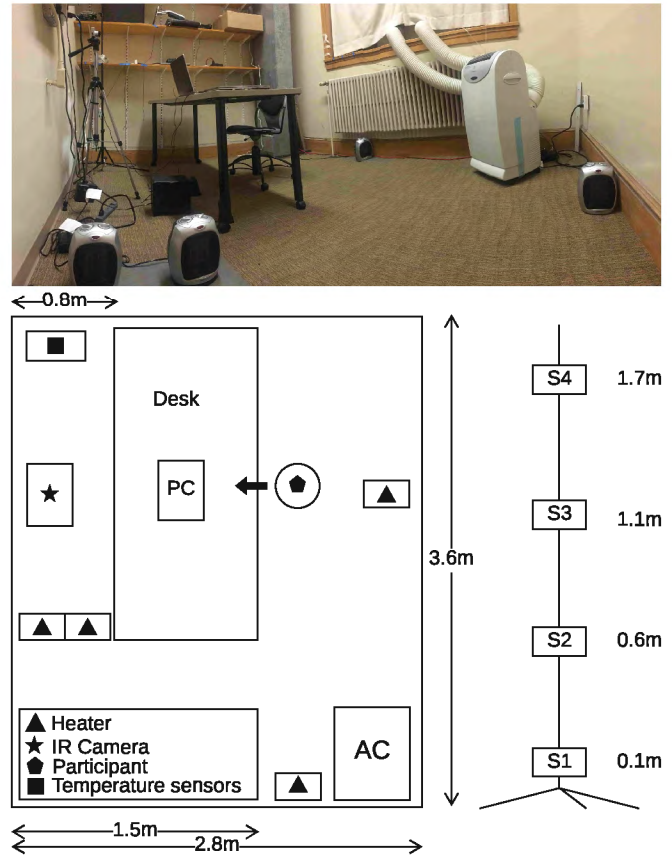


Figure 4: Top: Experiment lab Photo; Bottom Left: Top view of Lab and sensors' position. The participant was facing the direction with the arrow; Bottom Right: Room temperature sensors in different heights

3.6 Conditions

Each participant was randomly assigned to one of two temperature conditions: neutral to warm (25°C to 30°C), and warm to neutral (30°C to 25°C). By randomizing the thermal conditions, we avoid the potential confound that students' performance changed in different sessions not due to thermal comfort but due to other factors related to time, e.g., fatigue. If the participant was in the neutral to warm condition, the room temperature in the adaptation session was maintained at 25°C until the end of the first tutorial session; it was then increased to 30°C in the second tutorial session and was maintained at this level until the end of the third tutorial session. See Figure 5.

3.7 Data collection

Using the sensors, we collected several kinds of data from each person: (1) Video from the web-camera (at 30 fps); (2) Infrared images (1 every 21 minutes); (3) room temperature, CO₂, and relative humidity (1 measurement every minute); (4) body temperature (1 every minute for each sensor); (5) each participant's start/end times of each tutorial video, quiz, and survey; (6) each participant's quiz scores.

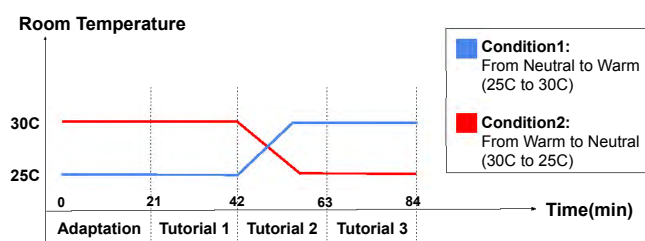


Figure 5: The change of room temperature in different conditions

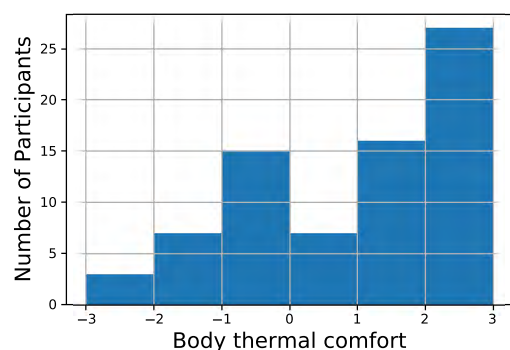


Figure 6: Histogram of thermal comfort in our experiment

4. ANALYSIS

Our analysis was focused on two questions: (1) what is the relationship between thermal comfort, temperature, and learning? (2) How can we use the various sensors to estimate participants' self-reported thermal comfort automatically?

4.1 Impact of room temperature on thermal comfort

In our experiment, the range of the room temperature was from 25°C to 30°C. This was not a huge change in the temperature. One of our goals was to assess whether this magnitude of temperature change could influence body thermal comfort. As defined in the thermal comfort survey that we used [3], the range of thermal comfort was from -3 to 3, where -3 means "very uncomfortable" and 3 means "very comfortable". Based on the histogram of body thermal comfort in our experiment in Figure 6, we see that the participants rarely (10 total votes) considered their thermal comfort to be highly uncomfortable (a rating of -3, -2). This indicated that our setting of the experiment was relatively comfortable for most of the participants. Did the modest temperature changes induced during the experiment impact participants' thermal comfort? To investigate, we considered models including either linear or quadratic terms for room temperature (computed as the average of the temperature sensors at different heights). The quadratic model did not give a stat. sig. better model fit, and hence we used a linear model; see Figure 7. The Pearson correlation between the model's predictions and self-reported thermal comfort scores was $r = -0.436$, $p < 0.001$, i.e., within the temperature range of our experiment, higher temperature resulted

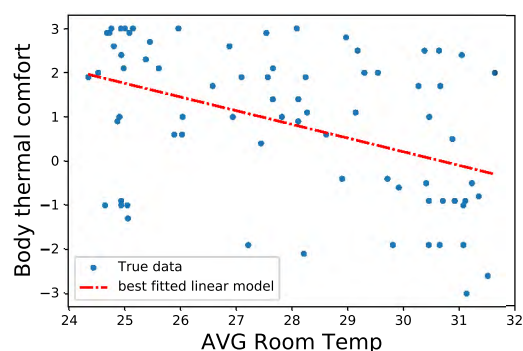


Figure 7: Thermal comfort VS Avg room temperature

in lower thermal comfort. Based on the estimated regression coefficient, increasing the room temperature by one degree in our temperature range results in a reduction of thermal comfort by 0.32. Note that we also tried modeling thermal comfort and temperature (linearly) with a participant-specific offset as a random effect and obtained similar results.

4.2 Relationship between thermal comfort, learning, and time

After showing the change of room temperature in our experiment could influence the participants' thermal comfort, we assessed whether thermal comfort was related to participants' performance in the learning task. A scatter-plot of the quiz scores versus self-reported thermal comfort scores is shown in Figure 8. Neither the Pearson nor the Spearman correlations between quiz score and thermal comfort were significant. However, after visually examining the scatter-plot, we noticed a slight "inverted U" shape; this has also been noted in prior work [29, 28]. This shape indicates that when the participants felt too comfortable or too uncomfortable, their quiz score were lower; when the thermal comfort state was in the middle, their quiz score was higher. We found some support for this hypothesis in our data: the Spearman correlation between the *square* of self-reported thermal comfort and quiz score was negative ($r = -0.235$) and statistically significant ($p = 0.0042$). The quadratic model of self-reported thermal comfort gives a stat. sig. better fit than the linear model (likelihood ratio test, $p = 0.002$).

To explore this more rigorously by accounting for repeated measures, we also used a mixed-effect model with a random effect to model an offset for each unique participant. Due to different tutorial videos having different difficulties, we also considered the video_id as the random effect. We studied the relationship between thermal comfort and quiz score within each of the three tutorial session (1, 2, 3) separately. To our surprise, in the first two tutorial session, the impact of the square of the body thermal comfort (i.e., TC^2) was not significant ($p > 0.05$). However, in the last (third) session, the impact was negative and stat. sig. ($p = 0.013$). The estimated magnitude was that a change in 1 level of thermal comfort decreases the quiz score by 0.2 points (the maximum score was 6 points). A possible interpretation is

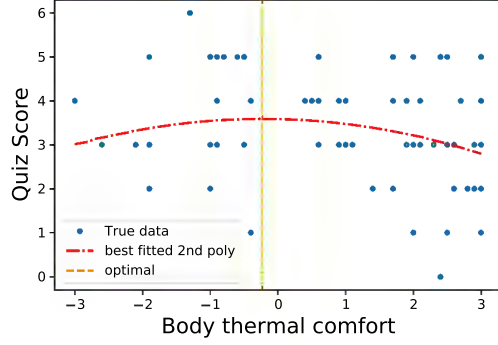


Figure 8: Thermal comfort VS Quiz score

Table 2: Effect size (Cohen’s f^2) of TC^2 in each tutorial session

Session No.	Effect size
1	0.007
2	0.044
3	0.308

that, as time went on, the participants might feel more tired or bored. At first, they could force themselves to focus on the tutorial videos and answer questions. However, when they became fatigued or bored, an uncomfortable thermal comfort might start to show its influence. See Table 2 for the effect size (calculated based on the marginal R^2) in each tutorial session.

4.3 Relationship between thermal comfort and sleepiness

The survey that each participant completed after every tutorial session contained questions not just about thermal comfort, but also about how sleepy they felt. The values ranged from -3 (very sleepy) to +3 (very alert). The correlation between thermal comfort and sleepiness was positive (0.32) and stat. sig. ($p = 0.0084$).

4.4 Relationship between engagement and learning

To explore whether the perceived level of student engagement, as judged by an external observer, was related to students’ learning, we manually labeled video frames from each participant’s face video. We extracted 1 frame every 20 seconds for each of the 3 tutorial sessions of all the participants. These pictures were labeled for the appearance of ‘engagement’ following the definitions in [36]. Level 1 is “not engaged”, level 2 is “nominally engaged”, level 3 is “engaged”, and level 4 is “very engaged”; see Figure 9 for a representative image of each label. During labeling, the images were randomized over time and also over participants; hence, the engagement scores were unbiased w.r.t. participants’ self-reported thermal comfort. We averaged the engagement for each participant per each of the three tutorial sessions, and then used a mixed effect model to analyze the relationship between quiz score and engagement. The participant_id was still the random effect. Since we had a prior hypothesis

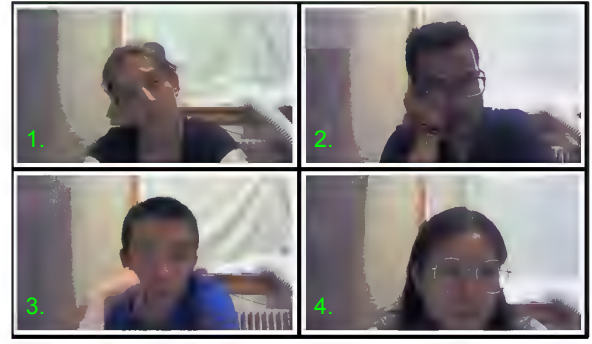


Figure 9: Participants in different engagement levels.

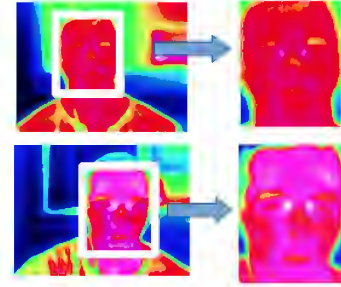


Figure 10: Manually cropped face for infrared images. Top: face when thermal comfort is -0.6. Bottom: face when thermal comfort is 2.7.

that engagement was positively correlated with learning, we used a 1-tailed t-test. The result showed that this positive correlation was significant ($p = 0.032$).

5. AUTOMATIC DETECTION OF THERMAL COMFORT

The primary method of estimating thermal comfort is via self-report on a survey. Might there be an automated way of obtaining this information that is less intrusive and gives higher temporal resolution? This could be useful to advance research on the IEQ and learning. Moreover, it could also set the stage for smart learning environments in which localized ventilation, heating, and cooling systems can optimize the thermal comfort for each learner. With these goals in mind, we explored several approaches to automatically estimating thermal comfort using the different sensors we deployed in our experiment.

5.1 Infrared camera

Per participant, 3 IR images were collected (one per tutorial session). From each IR image, we manually cropped the face for infrared images from IR camera and calculated the average face temperature for each tutorial session. For each IR image, we cropped the face between two ears for width, and from forehead to chin for length; see Figure 10. We then calculated the mean temperature within the face region and used it to predict thermal comfort. Using a mixed-effect model (with participant_id as a random effect), we found that the correlation between the face temperature, as computed from the calibrated IR image, and thermal comfort

Table 3: Skin Temp. VS Thermal comfort

Sensor	Pearson Correlation	p-value
D	-0.273	0.018
K	-0.174	0.136
O	-0.186	0.11
Q	-0.28	0.015

was -0.34 ($p = 0.0029$). In other words, a hotter face was associated with lower thermal comfort.

5.2 Skin sensors of body temperature

We averaged the skin temperature from 4 skin sensors for each tutorial session. The correlations between thermal comfort and averaged skin temperature are shown in Table 3.

With statistical significance, the correlations of the skin temperature at position D and Q indicated that they had a negative correlation with body thermal comfort. These two correlations also remained significant when we applied the mixed-effect model and set participant_id as random effect.

5.3 Web camera

Even though the results of skin sensors and infrared cameras showed that we could use them to detect thermal comfort, we were still interested in whether an ordinary (visible light) web camera can be used to detect thermal comfort. In contrast to skin sensors, web cameras are less intrusive – they require no skin contact or medical tape. In contrast to IR cameras, they are less expensive and more widely available.

While one could consider a “black box” approach such as a CNN-LSTM in which all the pixels of an entire video segment is used to predict thermal comfort, the relatively small size of our dataset ($n = 24$) makes this approach difficult. Instead, we investigated whether the much lower-dimensional feature representation of facial expressions can reveal a person’s thermal comfort. For example, we reported above that sleepiness is associated with thermal comfort, and this might be revealed in a person’s facial expression; this approach was used in [33] to detect drowsiness when driving a car.

After watching the videos, our subjective impression was that predicting thermal comfort from the face was very difficult. In the temperature range of our experiment setting, the facial expressions in different temperature condition did not vary greatly. Nevertheless, we tried three approaches: (1) estimate thermal comfort directly from the average facial features values extracted from OpenFace [4] over the time series of face images; (2) estimate thermal comfort from a Gabor-filtered time series of facial features; and (3) train a recurrent neural network to analyze the raw time series.

5.3.1 Individual face movements

From each frame in each 10-minute video sequence just prior to the self-reported thermal comfort survey of each tutorial session of each participant, we used OpenFace to extract the facial action units (AUs 1, 2, 4, 5, 6, 7, 9, 10, 12, 14, 15, 17, 20, 23, 25, 26, 45). In addition, we also calculated the size of the face – this could be useful for determining

**Figure 11: Landmarks from OpenFace**

if the participant leaned toward or away from the camera. Next, we extracted the head pose. Finally, we computed the distance between the eye-lids – this could give some measure of drowsiness.

For the left eye, we first calculated the central point of landmark 37 and 38, the central point of landmark 41 and 40, and then, calculated the distance between these two central points. For the right eye, we calculated the distance used landmark 43, 44, 47 and 46 as the same approach as the left eye. The eye-lid distance was the mean of the left distance and the right distance. We also estimated the size of the face box as an indication of whether a person was leaning towards or away from the camera: we first calculated the central of landmark 19 and 24, and then calculated the distance between the central and landmark 8, and also the distance between the landmark 0 and 16. The final face size was the product of the two distance. See Figure 11.

Using the above feature set, we examined the Pearson correlation between each mean feature value (averaged over each 10-minute time series) and self-reported thermal comfort. Only two features were stat. sig. correlated: AU 6 (Pearson $r = 0.244$, $p = 0.038$; see Figure 12) – cheek raiser – and the eye-lid distance, calculated by the landmarks on the eyes, was also correlated to thermal comfort with significant (Spearman $r = -0.27$, $p = 0.02$). The latter correlation suggests that smaller eye opening is associated with larger thermal comfort; this is consistent with the notion that thermal comfort that is “too high” may cause people to become sleepy.

5.3.2 Gabor filtered time series

A 1-D (temporal) Gabor filter is a complex-valued band-pass filter, with a specifiable center frequency and bandwidth, whose impulse response is local in both time and frequency; an example of the real component of one filter is shown in Figure 13. Gabor filters have been applied to var-



Figure 12: Example of AU 6 (<https://www.cs.cmu.edu/~face/facs.htm>)

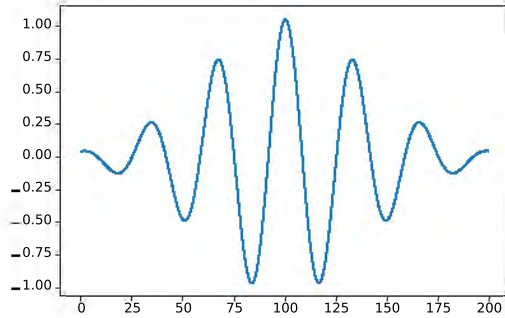


Figure 13: One example of real gabor filter. Frequency: 3.0; bandwidth: 0.9492

ious facial expression recognition tasks [33] and can capture certain patterns of a raw time series. For instance, they can capture wave-like patterns such as repeated blinking or eye closure. Here, we explored whether they could be helpful for predicting thermal comfort.

We applied Gabor filter to the AUs, face size, head pose, and eye-lid distance features. The frequency was selected from {8.0, 7.0, 6.0, 5.0, 4.0, 3.0, 2.25, 1.6875, 1.2656, 0.9492, 0.7119, 0.5339, 0.4005, 0.3003, 0.2253, 0.1689, 0.01, 0.} and the bandwidth was selected from the same set without 0. Thus, 918 filters (the combination of 18 frequencies, 17 bandwidths and real, imaginary and energy Gabor filters) were applied to each AU and head features, which was the same filter bank as [33]. We used forward feature selection to pick the top 5 filtered features and then used linear regression on these top 5 features to predict thermal comfort. However, even the best Pearson correlation was very low ($r = 0.02$), suggesting that this approach had limited predictive power.

5.3.3 Recurrent neural networks

Recurrent neural networks such as LSTM and GRU, are powerful models for dealing with time series. We explored whether a GRU (Gated Recurrent Unit) network can analyze the facial expression series to estimate thermal comfort. We trained a GRU model from the features extracted using OpenFace described above using leave-one-person-out cross-validation to measure accuracy of the approach. Hyper-parameters were selected from the sets {learning rate: {0.0001, 0.0005, 0.001}, hidden units: {8, 16, 32}, epoch: 50, optimizer: {Adam, SGD}. For each fold, we randomly selected 5 participants as the validation set (for hyperparameter validation), and the remaining 18 participants as the training set. Training every 5 epochs, the model would be applied to validation set and test set.

After tuning the hyper-parameters on the validation set, the best combination was {learning rate: 0.0005, hidden units: 32, epoch: 15, optimizer: Adam}. The average (over all 24 folds) correlation between predicted and actual thermal comfort scores was 0.248; the result was statistically significant ($p = 0.0425$, Wilcoxon signed-rank test). We note, however, that this result is no larger than the magnitude of the correlation between the eye-lid distance and thermal comfort reported above.

6. DISCUSSION AND CONCLUSION

We conducted an experiment in to investigate the relationship between thermal comfort and students' performance in a computer-based learning task in the classroom. We also explored different sensors and predictive models to measure thermal comfort automatically.

Key results: 1) Changing the room temperature by a few degrees Celsius could stat. sig. impact students' self-reported TC; (2) Our experimental data provide evidence that learning is optimal when thermal comfort is neither too high nor too low (inverted U relationship), corroborating prior work. However, we also found a more nuanced relationship than had been identified in prior literature: the impact of thermal comfort on learning was stronger during the third tutorial session (later in time) compared to the first two sessions. (3) Engagement, as labeled by an external observer, was correlated with learning. (4) Thermal comfort can be predicted from the face temperature using an IR camera. (5) Facial expression, at least in the ways we analyzed it, carries only limited information about thermal comfort.

Future work: Given a larger video dataset of face images and associated self-reported thermal comfort scores, we could explore more powerful prediction models that directly predict thermal comfort from the face pixels. This might offer more powerful information than the facial expression estimates from OpenFace.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant Nos. #1551594 and #1822768.

7. REFERENCES

- [1] Y. Al Horr, M. Arif, A. Kaushik, A. Mazroei, M. Katafygiotou, and E. Elsarrag. Occupant productivity and office indoor environment quality: A review of the literature. *Building and environment*, 105:369–389, 2016.
- [2] M. Arif, M. Katafygiotou, A. Mazroei, A. Kaushik, E. Elsarrag, et al. Impact of indoor environmental quality on occupant well-being and comfort: A review of the literature. *International Journal of Sustainable Built Environment*, 5(1):1–11, 2016.
- [3] ASHRAE. Standard 55-2004. thermal environmental conditions for human occupancy. *American Society of Heating, Refrigerating and Air-Conditioning Engineers*, 2004.
- [4] T. Baltrušaitis, A. Zadeh, Y. Chong Lim, and L.-P. Morency. Openface 2.0: Facial behavior analysis toolkit. 2018.
- [5] P. Barrett, F. Davies, Y. Zhang, and L. Barrett. The

- impact of classroom design on pupils' learning: Final results of a holistic, multi-level analysis. *Building and Environment*, 89:118–133, 2015.
- [6] P. Barrett, Y. Zhang, J. Moffat, and K. Kobbacy. A holistic, multi-level analysis identifying the impact of classroom design on pupils' learning. *Building and environment*, 59:678–689, 2013.
- [7] G. S. Brager and R. J. De Dear. Thermal adaptation in the built environment: a literature review. *Energy and buildings*, 27(1):83–96, 1998.
- [8] S. Choi, D. A. Guerin, H.-Y. Kim, J. K. Brigham, and T. Bauer. Indoor environmental quality of classrooms and student outcomes: A path analysis approach. *Journal of Learning Spaces*, 2(2):2013–2014, 2014.
- [9] R. De Dear and G. S. Brager. Developing an adaptive model of thermal comfort and preference. 1998.
- [10] P. V. Dorizas, M.-N. Assimakopoulos, and M. Santamouris. A holistic approach for the assessment of the indoor environmental quality, student productivity, and energy consumption in primary schools. *Environmental monitoring and assessment*, 187(5):259, 2015.
- [11] P. Fanger. Moderate thermal environments determination of the pmv and ppd indices and specification of the conditions for thermal comfort. *ISO 7730*, 1984.
- [12] P. O. Fanger et al. Thermal comfort. analysis and applications in environmental engineering. *Thermal comfort. Analysis and applications in environmental engineering.*, 1970.
- [13] A. Gilavand. Investigating the impact of environmental factors on learning and academic achievement of elementary students. *Health Sciences*, 5(7S):360–369, 2016.
- [14] U. Haverinen-Shaughnessy, D. Moschandreas, and R. Shaughnessy. Association between substandard classroom ventilation rates and students' academic achievement. *Indoor air*, 21(2):121–131, 2011.
- [15] F. Jazizadeh and W. Jung. Personalized thermal comfort inference using rgb video images for distributed hvac control. *Applied Energy*, 220:829–841, 2018.
- [16] J. Jiang, D. Wang, Y. Liu, Y. Xu, and J. Liu. A study on pupils' learning performance and thermal comfort of primary schools in china. *Building and Environment*, 134:102–113, 2018.
- [17] W. Jung and F. Jazizadeh. Vision-based thermal comfort quantification for hvac control. *Building and Environment*, 2018.
- [18] S. Kaltwang, O. Rudovic, and M. Pantic. Continuous pain intensity estimation from facial expressions. In *International Symposium on Visual Computing*, pages 368–377. Springer, 2012.
- [19] K.-i. Kameda, S. Murakami, K. Ito, and T. Kaneko. Study on productivity in the classroom (part 3) nationwide questionnaire survey on the effects of ieq on learning. *Clima 2007 WellBeing Indoors*, 2006(Part 3), 2007.
- [20] M. Lee, K. Mui, L. Wong, W. Chan, E. Lee, and C. Cheung. Student learning performance and indoor environmental quality (ieq) in air-conditioned university teaching rooms. *Building and Environment*, 49:238–244, 2012.
- [21] A. Lipczynska, S. Schiavon, and L. T. Graham. Thermal comfort and self-reported productivity in an office with ceiling fans in the tropics. *Building and Environment*, 135:202–212, 2018.
- [22] S. Liu, S. Schiavon, A. Kabanshi, and W. W. Nazaroff. Predicted percentage dissatisfied with ankle draft. *Indoor air*, 27(4):852–862, 2017.
- [23] W. Liu, Z. Lian, Q. Deng, and Y. Liu. Evaluation of calculation methods of mean skin temperature for use in thermal comfort study. *Building and Environment*, 46(2):478–488, 2011.
- [24] G. C. Marchand, N. M. Nardi, D. Reynolds, and B. Pamoukov. The impact of the classroom built environment on student perceptions and learning. *Journal of Environmental Psychology*, 40:187–197, 2014.
- [25] B. Pavlin, G. Pernigotto, F. Cappelletti, P. Bison, R. Vidoni, and A. Gasparella. Real-time monitoring of occupants' thermal comfort through infrared imaging: A preliminary study. *Buildings*, 7(1):10, 2017.
- [26] S. A. Samani and S. A. Samani. The impact of indoor lighting on students' learning performance in learning environments: A knowledge internalization perspective. *International Journal of Business and Social Science*, 3(24), 2012.
- [27] I. Sarbu and C. Pacurar. Experimental and numerical research to assess indoor environment quality and schoolwork performance in university classrooms. *Building and Environment*, 93:141–154, 2015.
- [28] O. Seppanen, W. J. Fisk, and Q. Lei. Room temperature and productivity in office work. 2006.
- [29] O. A. Seppänen and W. Fisk. Some quantitative relations between indoor environmental quality and work performance or health. *Hvac&R Research*, 12(4):957–973, 2006.
- [30] N. Srinivasan. Progress in brain research: Attention. 2009.
- [31] V. Todea. Guide for psycho diagnosis laboratory. *Timisoara: Artpress Publishing House (in Romanian)*, 2008.
- [32] S. Turkay and S. T. Moulton. The educational impact of whiteboard animations: An experiment using popular social science lessons. In *Proceedings of the 7th International Conference of Learning International Networks Consortium (LINC)*. Cambridge, MA, USA, pages 283–91, 2016.
- [33] E. Vural, M. Bartlett, G. Littlewort, M. Cetin, A. Ercil, and J. Movellan. Discrimination of moderate and acute drowsiness based on spontaneous facial expressions. In *2010 20th International Conference on Pattern Recognition*, pages 3874–3877. IEEE, 2010.
- [34] D. Wang, H. Zhang, E. Arens, and C. Huizenga. Observations of upper-extremity skin temperature and corresponding overall-body thermal sensations and comfort. *Building and Environment*, 42(12):3933–3943, 2007.
- [35] P. Wargocki and D. P. Wyon. The effects of moderately raised classroom temperatures and classroom ventilation rate on the performance of schoolwork by children (rp-1257). *Hvac&R Research*, 13(2):193–220, 2007.

- [36] J. Whitehill, Z. Serpell, Y.-C. Lin, A. Foster, and J. R. Movellan. The faces of engagement: Automatic recognition of student engagement from facial expressions. *IEEE Transactions on Affective Computing*, 5(1):86–98, 2014.
- [37] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. Freeman. Eulerian video magnification for revealing subtle changes in the world. 2012.
- [38] Z. S. Zomorodian, M. Tahsildoost, and M. Hafezi. Thermal comfort in educational buildings: A review article. *Renewable and sustainable energy reviews*, 59:895–906, 2016.

The Influence of School Demographics on the Relationship Between Students' Help-Seeking Behavior and Performance and Motivational Measures

Shamya Karumbaiah

Penn Center for Learning Analytics

3700 Walnut St

Philadelphia, PA 19104

+1 (877) 736-6473

shamya@upenn.edu

Jaclyn Ocumpaugh

Penn Center for Learning Analytics

3700 Walnut St

Philadelphia, PA 19104

+1 (877) 736-6473

jlocumpaugh@gmail.com

Ryan S Baker

Penn Center for Learning Analytics

3700 Walnut St

Philadelphia, PA 19104

+1 (877) 736-6473

ryanshaunbaker@gmail.com

ABSTRACT

Demographic information often proves useful for finding subpopulations in educational data. Unfortunately, it is often not collected in the log files of online learning systems, which serve as one of the primary sources of data for the Educational Data Mining community. Recent work has sought to address this issue by investigating school-level differences in demographics, which can be used to discover trends in data where individual-level variation may be difficult or impossible to acquire. In this study, we use this approach to investigate the effect of demographic patterns on hint usage in an elementary level mathematics system, comparing this use to performance and motivational measures. In doing so, we expand upon the research into help-seeking behaviors, which typically takes a cognitive approach. Our results suggest the need to better understand what social factors are most likely to motivate help-seeking behaviors, particularly since research on their effectiveness has been mixed.

Keywords

Help-seeking, Demographics, Math Learning, Math Identity, Self Concept, Self Regulated Learning

1. INTRODUCTION

Many studies into complex constructs like motivation, interest, and engagement involve either small-scale experiments or larger convenient samples of middle-class, undergraduate students (see discussion in [21]), which can make it difficult to determine the extent to which these findings will generalize to new populations of students. This is often due to the practical constraints of research projects involving the budget, recruitment, accessibility and time required to acquire the level of detail used in these studies. However, this trade-off sometimes means that conclusions are not replicated across broad demographic contexts. Conversely, Educational Data Mining (EDM) researchers often have larger sample sizes than are seen in experimental psychology, for example, but the source of typical EDM data (i.e., intelligent tutoring systems) often limits the practicality of obtaining demographic variables from individual students.

Shamya Karumbaiah, Jaclyn Ocumpaugh and Ryan S Baker "The Influence of School Demographics on the Relationship Between Students' Help-Seeking Behavior and Performance and Motivational Measures" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 99 - 108

Beyond practicality (e.g., the ease of acquiring log-file data on student interactions compared to student demographic data), there are sometimes other legitimate concerns, including those related to student privacy. For example, even when a partner school or university has documented the demographics of individual students, their release to a researcher increases the risk of potentially re-identifying students, particularly in rural parts of the country where the analysis of say, the seven children of a minority ethnic group in a small school narrows the potential matches for sensitive information considerably.

Yet considerable research shows that demographic factors are often related to differences in educational outcomes more generally (see [14]) and to constructs related to motivation more specifically [50, 50, 44]. This suggests that researchers in the EDM community should make greater efforts to overcome the challenges involved in collecting demographic data in order to ensure population validity (e.g., [29]). As such, some researchers within the EDM community have sought to extend student learning models to include information from the broader context, building models at the class, school-, school-cluster level instead of just the student-level [47, 31]. [49] used school-level demographics and students' prior performance to cluster schools into groups, improving model performance.

This study uses this approach to investigate hint-usage, incorporating the broader demographic context into the investigation while also answering a call to pay greater attention to the social factors influencing help seeking behaviors after a recent review of the literature found that their effectiveness was highly variable [15]. We do so in the context of Reasoning Mind—an Intelligent Tutoring System for elementary mathematics—where we explore how readily-available school-level demographics might reveal how hint usage correlates to measures of student performance and motivation (i.e., mathematics self-concept).

2. PRIOR WORK

Help - mostly in the form of on-demand, contextual, real-time hints - is a common feature in most Intelligent Tutoring Systems (ITSs) [46] and has long been believed to foster emerging concepts and principles in a student's learning [7] and support struggling students during problem-solving [1]. Yet help-seeking behaviors are not always beneficial [1, 2, 4]. While much of the prior work on help-seeking in ITSs has focused strictly on its cognitive effects, other research suggests that social factors may influence these patterns.

2.1 Help-Seeking: A Cognitive Lens

The literature on help-seeking behaviors in ITS now stretches back over twenty years (see extensive review in [4]). As it quickly became apparent that the availability of hints did not ensure their effective use, work began to identify the factors that led to a positive relationship between help-seeking behaviors and student learning.

In one of the earliest studies, Anderson et al. [6], compared the use of explanatory hints and so-called bottom-out hints (which simply provided the student with the correct answer and found that neither hint type was correlated with learning. In part, this may have been due to selection bias. That is, hint usage is typically a sign of struggling students, who often do not make substantial learning gains (see discussion in [4]).

After early discoveries of a negative correlation between hint usage on student learning in one context [1], researchers began to develop a taxonomy of maladaptive help-seeking behaviors—including categories like help abuse (the overuse of help) and help avoidance (the underuse of help)—was also developed [13]. Most studies analyzed the effectiveness of hints by focusing on the relationship between help-seeking behavior(s) and student outcome(s), with some researchers emphasizing that the intentionality of help-seeking behavior makes it a good candidate for understanding students' self-regulated learning (SRL) strategies [4, 16].

A number of studies have attempted to identify the degree of help needed at any given moment (e.g., Koedinger & Aleven's [24] *assistance dilemma*), and experimental results have resulted in notable findings. For example, (1) on-demand hints led to greater learning gains than automatic hints in middle-school mathematics [32]; (2) hint content (goal feedback versus other kinds of feedback) is related to student learning in Geometry [27]; (3) hints about which step to try next to improve student learning in logic proof [36].

In general, the review of the literature suggests that increasing hint usage does not always lead to better domain-level learning [4]. However, the EDM literature on help-seeking in ITSs has produced research which aggregates into a complicated and contradictory narrative, including: (1) a negative association between hint usage and learning [2]; (2) a positive association between hint usage and learning [11, 48]; (3) a positive association between hint usage and learning only when time per hint level is considered [25] or when adaptive versus maladaptive help-seeking is differentiated [3]; (4) a positive association between time spent in bottom-out hints and learning [35]; (5) a negative association between the number of bottom-out hints used and learning [26]; (6) positive benefits for students but only when they have a medium level of skill [33]; (7) a negative association between help avoidance and learning early within practice [5] and on a transfer post-test [9].

In addition, individual differences in self-regulation were observed in how students process hints and how that impacts their performance [16]. Vaessen et al [45] found that students' achievement goals (mastery and performance goals) are closely related with their help-seeking and could be used to predict their strategies for help-seeking. Despite a considerable volume of research, the effectiveness of help-seeking remains an open question—and the clearest thing that we can say is that the relationship between hint usage and learning is complicated.

2.2 Help-Seeking: A Social Lens

While the role of social factors on help-seeking behaviors has not been the primary focus of the EDM community (see [4]), the social evaluation of help-seeking behaviors is well established in the literature. For instance, some learners may feel that asking for help is either a sign of incompetence or a challenge to their autonomy [39]. Likewise, Howley et al. [18] suggests that asking for help may trigger experiences of evaluation anxiety – the fear of being judged.

These kinds of concerns seem ripe for socio-cultural variation, and a few studies have begun to explore how these differences may emerge. For example, Tai et al. [38] increased students' help-seeking behaviors by changing the way they labeled those actions within the system. That is, they started by referring to the ITS as the students' teammate, and the designed the system so that students who needed help could choose to "work together" with the system. This adaption apparently reduced the ego-threat related to admitting a lack of knowledge (e.g., [39]) and improved student learning.

Other studies have specifically investigated demographic differences in help-seeking behaviors. Ogan et al. [30] found that the EDM models on effective help-seeking did not transfer well between countries (namely Costa Rica, the Philippines, and the USA). Likewise, Arroyo et al. [8] found that the effectiveness of different hint designs varied by gender. Specifically, girls benefited more from highly interactive hints, while boys did better with less interactive hints.

Thus, there is a need for more research to look at social factors while studying help-seeking. Such studies should focus on students' broader context to understand under what circumstances lead to desired student outcome. In this paper, we study students' help-seeking behavior in an online math tutor used in traditional classrooms during regular instruction. Given the context in which the students use this ITS, we focus on school as the social context and analyze the influence of school demographics on the relationship between student outcomes (math performance and math self-concept) and their help-seeking behavior. We aim to shift the focus of help-seeking research in the EDM community from purely cognitive factors to the contextual factors that might play a more prominent role than is assumed.

2.3 The Role of Demographics in Predicting Student Outcomes

This section summarizes prior work on the role of demographics in the student outcomes of interest in this study - math performance and math self concept.

2.3.1 Demographics and Math Performance

The literature addressing demographic differences in learning outcomes (at least in a U.S. context) is now so vast that it would be difficult to review even if it were limited to a single domain (e.g., mathematics). Once referred to as the achievement gap, more and more scholars are now discussing it in terms of an opportunity gap, as findings generally show that achievement patterns favor groups for whom the educational system was initially designed.

Scholars point out that reframing this discussion in terms of opportunities to learn emphasizes the need to address the environmental inadequacies that may occur. Childs [14] analysis shows, for example, that minority students are just as likely to value mathematics, but are less likely to attend schools where advanced mathematics classes are offered.

However, less tangible differences may also play a role. For example, if students' patterns of communication are different than those expected by educators (e.g., [19]), their attempts at help-seeking may not receive adequate uptake. Such experiences could discourage students from future help-seeking behaviors, although one could imagine that the ability to get help from an ITS could also mitigate this reluctance.

2.3.2 Demographics and Self-Concept

Demographic variables have also been shown to correlate with motivational constructs, like math self-concept. Math self-concept (sometimes used interchangeably with self-efficacy, although see [12]) has been found to be a predictor of various measures of achievement and career choice (see [13]). It has also been linked to motivational constructs, including achievement goal orientation, anxiety, and self-concept [34].

Early work proposed that self-efficacy was a product of a person's own accomplishments and the feedback they receive on their work [10, 43]; however, more recent studies have indicated that the source of self-efficacy may vary along demographic lines like gender and ethnicity [50, 50, 44]. For example, Klassen [23] investigation of self-efficacy among seventh grade students found that ethnic majority students followed Bandura's predictions, citing personal achievements as a source of self-efficacy, but ethnic minority students were more likely to cite social factors. Else-Quest, et al. [15] studied the intersection of gender, ethnicity, and achievement in 10th grade students from a large northeastern city and found that males reported greater math self-concept and expectation of success as compared females, but no gender differences across ethnic groups were found.

Other research on self-efficacy suggests that it is malleable and can be influenced by social agents [51], and there are significant efforts to understand how to support underrepresented groups, who may struggle against implicit stereotypes on top of normal learning struggles as their domain knowledge matures [37]. Previous research shows that scores on social identity ratings (e.g. gender and cultural identity ratings) peak when people are experiencing uncertainty [17]. This could suggest that students could become more susceptible to negative cultural stereotypes (e.g., [37]), particularly those related to STEM performance, during periods of confusion associated with learning, making help-seeking an important behavior to study for its associations with self-concept.

Given these findings, it seems likely that self-concept could vary not just by the demographics of individual students, but also based on how those demographics influence the cultural interactions at a school level. That is, in a school where larger numbers of students share a particular demographic characteristic, we might see help-seeking behaviors that emerge as a reflection of the practices more typical of that group.

3. DATA COLLECTION

3.1 Reasoning Mind

This study analyzes data from students using Reasoning Mind (RM) Foundations (Figure 1), an intelligent tutoring system for elementary mathematics, produced by Imagine Learning. It currently serves over 100,000 U.S. students annually. The majority of these students are in Texas, but they represent a range of traditionally underrepresented populations across rural, urban, and suburban schools. Key components of this system include socio-technical innovations, including those that are designed to

directly support teachers [21] and those that are designed to mimic other social experiences in the classroom, including both virtual peers and the signature pedagogical agent, known as the Genie, that guides students in their learning.

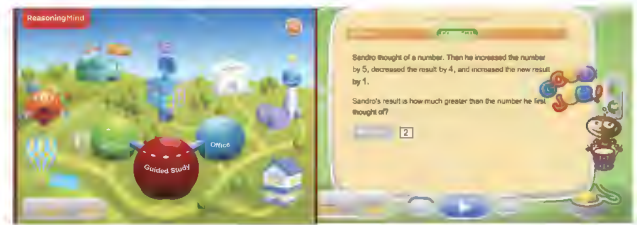


Figure 1. Reasoning Mind Foundations home screen (left) and an example problem (right)

In this blended environment, students learn through self-paced problem solving, interactive explanations, and skill-based games. Problem sets are classified into three groups based on increasing levels of difficulty: (1) A-level problems on fundamental skills; (2) B-level (optional) problems on a combination of skills; and (3) C-level (optional) problems on higher-order thinking skills. Our past study [20] suggests a close relationship between inconsistencies in students' math performance and their math self-concept – the two student outcomes studied in this paper. Reasoning Mind Foundations is generally used in traditional classrooms. Teachers assign/unlock problem sets for students based on the topic of instruction. Past studies of Reasoning Mind Foundations have shown high student and teacher acceptance, increases in test scores, high time on task, and a positive affective profile [21].

3.2 Hints in Reasoning Mind

Hints are an integral part of Reasoning Mind Foundations. These are delivered only on student request and contains conceptual feedback intended to help students solve the problem. Figure 2 demonstrates a hint in the system for one of the basic A-level problem in Reasoning Mind Foundations. They are multi-level and do not always contain a bottom-out hint.

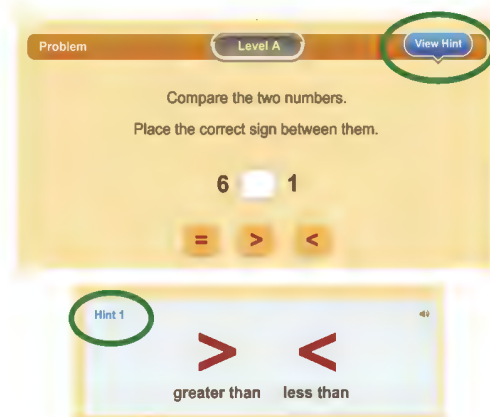


Figure 2. Top - Problem screen with a button to view hint (highlighted in green) Bottom - Hint displayed to the student when they request to view

3.3 Participant Schools

In order to ensure consistency in the type of data used to characterize schools, this study limits itself to Reasoning Mind

schools that fall under the purview of the Texas Education Agency's (TEA) and further filters out schools where less than 25 students were using the software to avoid noise in the correlations reported below. This resulted in data from 110 Texas schools across 25 school districts who used Reasoning Mind during the academic year 2017-2018 as part of their regular mathematics instruction. There are a total of 9,122 2nd through 5th grade students in this data (4,749 2nd graders, 1,964 3rd graders, 1,582 4th graders, and 827 5th graders). On average, there were 75 students using Reasoning Mind Foundations per school (min = 25; SD = 70) and 364 per school district (SD = 730), with one large urban district in Texas constituting the majority of our data, with 3,039 students across 62 schools.

Comprehensive log data captured student interactions with the system for the entire period, resulting in data for all 9,122 students. Surveys were administered once at the beginning and once at the end of the year to collect data on student math identity, resulting in complete surveys for 2,238 students.

4. DATA EXPLORATION

Considerable variation exists in the measures being analyzed in this study: help-seeking behaviors (i.e., hint usage), math performance, and pre- and post-year measures of math self concept.

4.1 Exploring Help-Seeking

From the interaction log data, we operationalize help-seeking behavior using as the number of hints used by a student in Reasoning Mind Foundations. As shown in Figure 3 (left), students in this study averaged less than 30 hint requests annually (mean = 27.01, SD = 55.72).

4.2 Exploring Math Performance

For the purposes of this paper, math performance is defined as the accuracy of student responses to A-level problems in Reasoning Mind Foundations. Accuracy on these problems, which represent the core curriculum within the software, is computed from the interaction log data. As presented in Figure 3 (right), student-level calculations show a mean of 0.77 (SD = 0.14).

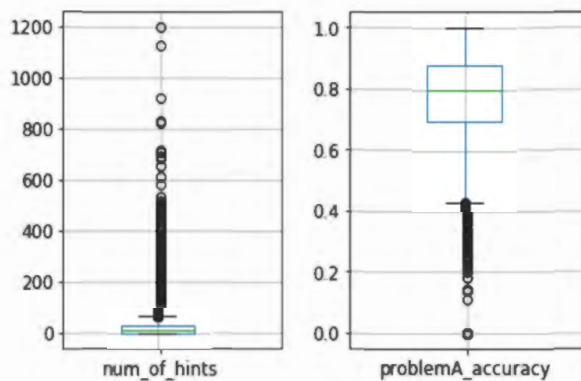


Figure 3. Distribution of the number of hints (left) and math performance (accuracy in A-level problems; right). The green line in the box indicates the median value.

4.3 Exploring Math Self Concept

Students' self concept in mathematics was measured using a five-item survey adapted from Marsh et al. [28]. This survey was

administered twice--once at the beginning of the academic year (pre) and once at the end of the academic year (post). The survey included questions like *Math just isn't my thing; Some topics in math are just so hard that I know from the start I'll never understand them*. Students took the survey voluntarily, and each item in the survey was answered with a four-point Likert scale.

This study analyzes survey responses from 2,238 students across 22 Texas schools. The distribution of students' responses is given in Figure 4 (self concept pre: mean = 2.64 standard deviation = 0.77; self concept post: mean = 2.44, standard deviation = 0.80). The internal consistency of these items was found to be satisfactory with a Cronbach's α of 0.74.

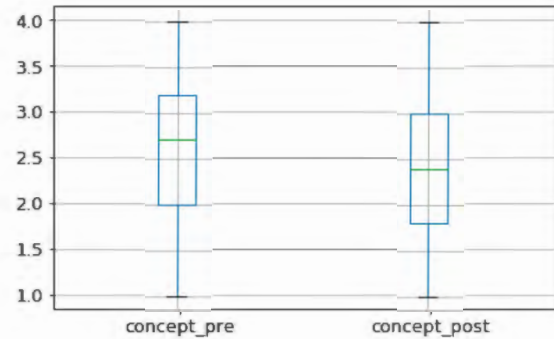


Figure 4. Distribution of number of pre and post measures of math self concept.

4.4 Exploring School-Level Differences

Next, we explored the school-level differences in student outcomes (math performance and self concept) and hint usage. As we can see in Figure 5 and Table 1, there is considerable variance in the variable aggregates (mean) across the schools, especially in hint usage and math performance.

Table 1. Mean and standard deviation (SD) of the school-level aggregates of the variable and outcomes

	Mean	SD
Hint Usage	24.5	21.3
Math Performance	0.78	0.04
Math Self Concept (Pre)	2.69	0.35
Math Self Concept (Post)	2.43	0.15

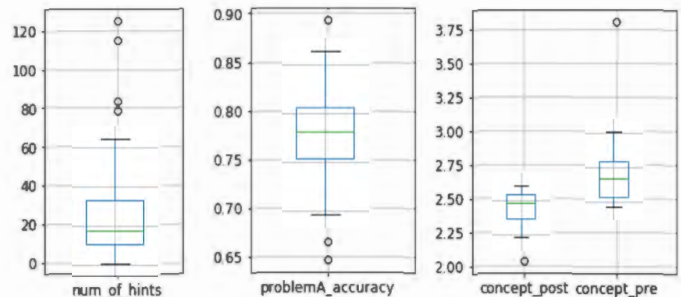


Figure 5. Distribution of school-level aggregates of the variable and outcomes.

4.5 Summarizing School-Level Demographics

We characterize the schools in our sample using demographics from the Texas Education Agency's (TEA) public data repository. These data capture the contextual factors that are likely to affect the school culture or climate and defines the social context in which students using RM Foundations.

Table 2. Mean and standard deviation (SD) of the school-level demographics. EcD - Economically Disadvantaged; LEP - Limited English Proficiency; SpEd - Special Education

	Mean	SD
Urbanicity (binary)	0.6	0.49
% EcD	78.3	16.6
% LEP	41.4	20.6
% SpEd	6.9	3.1

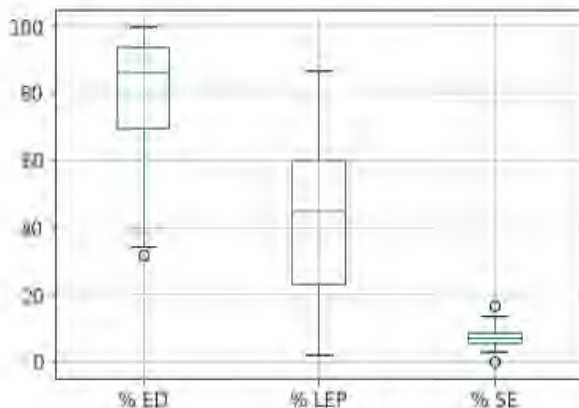


Figure 6. Distribution of percentages of school-level demographics for the 110 schools selected in this study. ED - Economically Disadvantaged; LEP - Limited English Proficiency; SE - Special Education

Table 2 summarizes the first set of school-level demographics obtained from TEA sources, including the percentage of students at the school who are classified as (1) Economically Disadvantaged (EcD), as (2) Special Ed (SpEd) or as (3) Limited English Proficiency (LEP), as well as (4) the urbanicity of the school. These terms are defined by the State of Texas as follows [40]. Students are classified as EcD if they qualify for free or reduced-price meals under the National School Lunch and Child Nutrition Program, and it is worth noting that a large proportion (avg = 40%) of Texas public school students qualify for this status [40]. SpEd classifications are given to students who qualify for services for cognitive, emotional, or physical disabilities. LEP status is conferred for students whose primary home language is not English and who also fail to meet proficiency standards as established by either an approved testing measure or by a Language Proficiency Assessment Committee (LPAC). Finally, the TEA classifies a school district as urban (or not) [41] based on whether its school district (a) is located in a county with a population of at least 960,000; (b) has the largest enrollment in the county or its enrollment is greater or equal to 70% of county's

largest district. As seen in Table 2 and Figure 6, we have a diverse set of schools along these dimensions.

We also considered school-level data on the percentage of students representing major ethnic/racial groups. As Table 3 shows, Hispanic students are by far the largest group in these schools (mean = 63.5%), followed by African American students (mean = 17.5%), White students (mean = 13.5%) and then Asian students (4.5%), but as Figure 7 the schools show considerable variance in terms of this composition. To avoid noisy results, this analysis considers only groups that constitute at least 5% of the student population: Hispanic, African American, White and Asian.

Table 3. Standard deviation (SD) of the school-level percentages of ethnicities. Categories constituting less than 5% of the data were excluded from further analysis.

	Mean	SD
% Hispanic	63.5	24.5
% African American	17.5	17.8
% White	13.1	16.2
% Asian	4.5	7.8
% American Indian*	0.36	0.4
% Pacific Islander*	0.04	0.1
% Two or More Races*	1	1

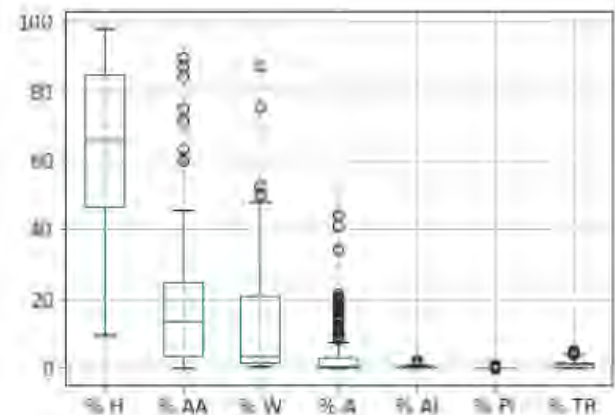


Figure 7. Distribution of percentages of school-level ethnicities for the 110 schools selected in this study. H - Hispanic; AA - African American; W - White; A - Asian; AI - American Indian; PI - Pacific Islander; TR - Two or races

5. ANALYSIS

Our data exploration (Section 4) suggests that help-seeking behavior, math performance, math self-concept, and demographics each vary by school. Thus, we conduct a two-step data analysis to explore how help-seeking behavior might differ based on student demographics, while controlling for performance and motivational measures.

In the first step, we determine how closely students' math performance (and self concept measures) correlate to their hint

usage, within each school, using Spearman ρ correlations due to non-normality in the data. That is, we produce three new measures for each student, the correlation between hint use and performance on A-level problems, the correlation between hint use and the pre-year survey of self concept, and the correlation between hint usage and the post-year survey of self concept.

In the next step, we determine whether the differences in these correlations are themselves correlated to school-level demographics. Note that in the first step, the unit of analysis for the correlations is the student, but in the second step, the unit of analysis is the school. We conduct two-tailed tests to report the significance levels.

6. RESULTS

6.1 Help-Seeking and Student Outcomes

Figure 8 summarizes the results for our study, showing the distribution of correlations across schools between students' hint usage and their math performance and math self-concept (taken once at the beginning (pre) and again at the end of the year (post)).

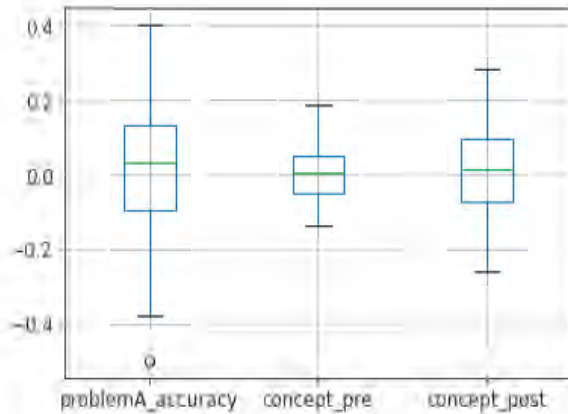


Figure 8. Distribution of correlations across schools between students' hint usage and outcomes.

6.1.1 Help Seeking and Math Performance

Clustering students by schools allows us to see that the relationship between hint usage and math performance differs in ways that might be missed if this aggregation were not used. This is true even when demographic descriptions are not used to describe the data.

Specifically, when student measures are aggregated at the school level, we see that the correlation between hint usage and math performance ranges from -0.39 to 0.40 ($SD = .18$). In contrast, when we do not aggregate students into school-level populations (instead treating them all as a single population), there is not a significant relationship between hint usage and math performance ($\rho = -0.008$, $p = 0.44$).

6.1.2 Help Seeking and Math Self Concept

Like math performance, math self concept also shows signs of sub-population differences. When students are aggregated into school-level populations, the correlations between hint usage and math self-concept show a relatively wide range.

For pre-year surveys, the correlation ranges from -0.14 (students with lower self-concept are most likely to use hints) to 0.19

(students with higher self concept are most likely to use hints), and an even wide range is found for post-year survey correlations (-0.27 to 0.30). In contrast, when the students in this data were treated as a single population, the correlations were non-significant ($\rho = -0.008$, $p = 0.442$) for pre and $\rho = -0.007$, $p = 0.77$ for post).

6.1.3 Summary of Help Seeking Variance

There is considerable variance in the school-level correlations between hint usage and student outcome measures ($SD = 0.18$ for math performance, $SD = .084$ for pre-year self concept, $SD = 0.118$ for post-year self concept). This variance indicates that students likely have different motivations for using hints, and they be more effectively used by some student populations than by others.

As seen in Figure 8, the median of the correlations is centered close to zero. For these schools, there is no association between hint usage on student outcomes. Figure 8 also shows that the distribution of these correlations is not skewed, meaning that hint usage is not universally positively or negatively associated with student outcomes across schools.

The schools at the tail ends of these distributions are interesting case studies. They represent the cases where hint usage has either a notably high positive correlation or a notably high negative correlation with our outcome measures. In schools where there is a high positive correlation between these variables, the use of hints appears to be beneficial, but the converse is true for those schools that have high negative correlations. As such, it becomes important to understand what demographics are involved in order to address any potential disparate impacts of the hint function in the system.

6.2 The Influence of School Demographics

School-level demographic variables help to capture some of the variance in the relationship between hint usage and the student outcomes measured in this study (math performance and math-self concept). These findings are summarized in Tables 4 and 5.

Table 4. Correlations between school-level demographics and the correlations resulted between students' math performance and interaction features. p-value in parenthesis. Significant correlations in bold.

	Correlation between number of hints and		
	Math performance	self concept Pre	self concept Post
Urbanicity	0.292 (0.002)	0.130 (0.564)	0.080 (0.729)
%EcD	0.256 (0.007)	0.182 (0.417)	-0.288 (0.205)
%LEP	0.314 (0.001)	-0.452 (0.035)	-0.565 (0.008)
%SE	-0.002 (0.982)	0.463 (0.030)	0.444 (0.044)

Table 5. Correlations between school-level ethnicity and the correlations resulted between students' math performance

and interaction features. p-value in parenthesis. Significant correlations in bold.

	Correlation between number of hints and		
	Math performance	Self concept Pre	Self concept Post
% Hispanic	0.094 (0.329)	0.123 (0.587)	-0.153 (0.507)
% African American	0.054 (0.579)	-0.260 (0.243)	-0.174 (0.451)
% White	-0.194 (0.042)	0.103 (0.647)	0.095 (0.683)
% Asian	-0.037 (0.703)	-0.071 (0.753)	-0.107 (0.644)

6.2.1 School-level Demographics, Help Seeking, and Math Performance

As Table 4 (above) shows, the relationships between hint usage and math performance differ significantly in terms of the school's urbanicity ($\rho = .292$, $p = .002$) as well as differences in the percentage of students who are economically disadvantaged (EcD; $\rho = .256$, $p = .007$) and limited English proficiency (LEP; $\rho = .314$, $p = .001$). Specifically, the association between higher hint usage and math performance is positive among students from urban schools than students from rural or suburban schools. Schools with higher percentage of students who are economically disadvantaged (EcD) or limited English proficiency (LEP), show the same trends. Conversely, rural and suburban schools show an inverse relationship between hint usage and math performance, suggesting that these students may be using hints ineffectively.

However, as Table 5 shows, other demographic categories that are often considered in educational research, namely ethnicity, are not particularly useful in this context. Schools with smaller populations of White students are more likely to show a positive relationship between hint use and math performance, whereas this relationship is more likely to be negative in schools with larger populations of White students. However, neither the percentage of African American students (which tends to be relatively small in the state of Texas and in this sample in particular) nor the percentage of Hispanic students (which tends to be quite large) is correlated with this relationship.

6.2.2 School-level Demographics, Help Seeking, and Math Self Concept

School-level demographics are less helpful in explaining the relationships between hint usage and math self concept. The relationships between hint usage and math self concept differ significantly in terms of the percentage of students with limited English proficiency ($\rho = -.452$, $p = .035$ for pre; $\rho = -.565$, $p = .008$ for post), and the percentage of students in special education ($\rho = .463$, $p = .030$ for pre; $\rho = .444$, $p = .044$ for post)). Specifically, in schools that serve a higher percentage of LEP students, there is a negative correlation with hint usage and self concept. Whereas, hint usage is more common among students with high self concept in schools that serve fewer LEP students. This finding is somewhat stronger for the end of year surveys than

the start of year surveys. The opposite pattern is shown among schools that serve a higher percentage of SpEd students. In these schools, there is a positive correlation between hint usage and self concept, where as that relationship is negative in schools that serve fewer SpEd students. This relationship is consistent across the start of the year and end of the year surveys.

Other demographic factors from Table 4 that were predictive of the relationship between help seeking and math performance, namely urbanicity and EcD, were not significant for the relationship between help seeking and math self concept. School-level descriptions of ethnicity (Table 5) also did not help to explain the variance between math self concept and hint usage.

7. DISCUSSION

7.1 Overview of Results

Hint-seeking behaviors have been a source of interest among EDM researchers since the early days of the field, yet understanding which hints are effective, to whom, and under what conditions remains a somewhat elusive task.

A large part of answering these questions likely lies in understanding what motivates a student to seek help. Ideally, we would like students to use these functions to improve their understanding of the material, but as these results show, students who are struggling do not always make use of available resources (e.g., in schools where low performers are not requesting as many hints).

However, within this data—which studies students in the same state using the same mathematics learning system—there are also schools where low-performing students are requesting lots of hints. If these students are benefiting from this hint usage, it is not measurable with the variables considered in this study. This finding suggests that the hints could be less effective at helping these particular students to learn the material.

At least part of this variance seems to be related to school-level demographics, but interestingly, the schools where hint usage appears to be most advantageous are those that enroll larger numbers of students who would typically be thought of as disadvantaged by the school system. That is, schools with fewer LEP students are more likely to have low performers who are requesting lots of hints. Schools with fewer students receiving free or reduced price lunch are more likely to have low performers who are requesting lots of hints. Schools in large urban centers are less likely to have low performing students who are requesting lots of hints.

The relationship between hint usage and self-concept is also complicated. Students in schools that serve more LEP students tend to show a negative relationship between self concept and hint usage. That is, those students who are unsure of themselves are asking for more hints (in those schools). However, in schools that serve more SpED students, the relationship between self concept and hint usage is negative. It is also possible that the smaller number of students sampled for self concept (compared to math performance) made it more difficult for these relationships to emerge.

Ethnic population differences were not particularly revealing in this study, and it is not entirely clear why. It is possible that, say, the LEP findings are strong enough to warrant further divisions to the subpopulations included in this study, a possibility that has not yet been explored in this data. However, it is also possible that some of the linguistic differences that influence classroom practices different ethnic groups within the United States (e.g.,

[19])—practices that may include figuring out how to ask for help—are less relevant in an online context like Reasoning Mind where the student is simply pressing button to request a hint.

Gender was not investigated in the current paper, as public schools generally have balanced gender distributions (as was the case in this dataset), leading to limited power to observe any difference that might exist. This leads to a more general point. It would be beneficial to analyze the impact of demographics at the student level, both to replicate the relationships seen here and to study whether students who are outliers in their own schools have different patterns. However, collecting student-level data is not always feasible, and this study has demonstrated that school-level aggregates can likely improve our efforts to better capture variance in help-seeking behaviors.

7.2 Implications to ITS Designers

One of the main implications of this paper to ITS designers is that a universal design that focuses on improving student outcomes while ignoring individual or group differences might not be a fair design consideration. Personalization in help-design has primarily focused on student cognition to provide “in-context” hints based on the pedagogical content. We suggest expanding the definition of “in-context” to include broader contextual factors that impact student outcomes.

To illustrate this, let’s take the example of LEP. As stated in Section 6.2, there is an inverse influence of help-seeking and the two student outcomes (performance vs. self concept). In schools with a higher percentage of limited English proficient students, higher hint usage is associated with high math performance but low math self-concept. On the other hand, in schools with more native English speakers, higher hint usage is associated with low math performance but high math self-concept. This is an interesting case of conflict for ITS designers to investigate further. Is the text-heavy nature of the hints contributing to this finding? Is it that while limited English proficient students use hints to improve on their math skills, the cognitive load in processing more verbal content is causing a negative impact on their self-efficacy? Such investigations could open up opportunities for design innovations to better support students. Would it help to use multiple representations (visual, auditory, symbols) and give autonomy to the students to make the choice? In summary, including school-level demographics to the analysis of complex constructs like help-seeking is an important step in appropriately situating the design decisions to the student context.

7.3 Limitations and Future Work

We acknowledge that there are other socio-cultural aspects that influence a student’s engagement and learning with an ITS. In the case of students’ help-seeking behavior, the perceptions of help-seeking within their classroom (peers, teachers) and outside (family, friends) can be influence student choices. While this paper focuses on broadly-defined school-level demographics, we believe that it would be beneficial to look at other influencers from the student’s social context. For instance, the pedagogical practices of the teacher in the math classroom could influence what students perceive as appropriate help-seeking.

Within Reasoning Mind Foundations, specifically, teachers are able to explicitly choose to design which problem types to assign to their students, in line with their pedagogical goals and perceptions of the appropriate level of difficulty. These choices likely reflect the classroom culture they are hoping to foster—including the degree to which they encourage students to attempt new skills and persevere in the face of challenges. There is an

opportunity to explore this data to study the impact of teacher choices on the relationship between help-seeking and student outcomes.

More broadly, the priorities of the school district and state might also impact the pedagogical choices made in schools. Teachers’ choices are influenced by public policy. Shortly after the completion of our data collection, Texas issued letter grades (A-F) [42] to its school districts based on a complex formula involving overall student performance on standardized exams, overall year-to-year improvement, and improvement for specific sub-groups. These ratings were generally lower in districts with higher rates of economically disadvantaged students, creating different degrees of pressure where demographics differ. The pressure of performing well (as measured by standardized tests), in many cases with limited resources, could influence what is being prioritized as the goal of math learning in these schools. While quantifying these factors to include in an analysis is not straightforward, these factors no doubt drive the type of differences that are seen between schools with different demographics.

8. CONCLUSION

Self-regulation is an important aspect of successful learning. Intelligent Tutoring Systems like Reasoning Mind Foundations provide a unique opportunity for students to practice self-regulation by taking control over their choices in the learning environment. Help-seeking is a particularly relevant SRL process within this type of learning system, given the prominence of hints in ITSs. In this paper, we demonstrate that school-level demographics can have a significant influence on the relationships between students’ help-seeking behavior and student outcomes. In doing so, we question the prevailing assumption that complex constructs like help-seeking can be considered without also considering student context. This calls for greater consideration within our field of social, cultural, and economic influencers outside cognition.

Amidst the mixed results from empirical studies on the effectiveness of hints, Aleven and colleagues [4] continue to recommend the use hints in ITSs and suggest making four key methodological distinctions when studying interventions designed to promote help-seeking - (1) effects on learning in the same learning environment versus a new environment; (2) effects on current learning versus future learning; (4) effects on learning in the same domain versus another; (3) effects on SRL process versus domain-level learning. We propose to extend upon the list of these methodological considerations, suggesting that researchers also (5) explore the effects of help-seeking designs in one demographic context versus another. We make this proposal while fully understanding both the practical challenges elaborated in the introduction and the definitional issues elaborated in Section 7. However, as we can see that such demographic effects are present even within a single U.S. state (albeit one of the larger and more diverse U.S. States), it is worth considering the ways in which different groups of people may attach different meanings to the behavior of help seeking. In particular, research should consider the ways in which help-seeking might be interpreted as an imposition or as an admission of failure, since, as we discussed in Section 2, these interpretations likely vary from one culture to another. By considering demographics in our research on help-seeking—and on SRL in general—we increase the likelihood that our findings will apply to the full diversity of learners using ITSs and related systems today.

9. ACKNOWLEDGMENTS

Our thanks to the NSF (Cyberlearning Award #1623730) for sponsoring this project, and our thanks to Matthew Labrum and Wanjing-Anya Ma for their support in data preparation.

10. REFERENCES

- [1] Aleven, V., & Koedinger, K. R. (2000). Limitations of student control: Do students know when they need help? In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), *Proceedings of the 5th International Conference on Intelligent Tutoring Systems, ITS 2000* (pp. 292–303). Berlin: Springer. Aleven et al., 2006
- [2] Aleven, V., & Koedinger, K. R. (2001). Investigations into help seeking and learning with a Cognitive Tutor. In R. Luckin (Ed.), *Papers of the AIED-2001 Workshop on Help Provision and Help Seeking in Interactive Learning Environments* (pp. 47–58).
- [3] Aleven, V., McLaren, B. M., Roll, I., & Koedinger, K. R. (2006). Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *International Journal of Artificial Intelligence in Education*, 16, 101–128.
- [4] Aleven, V., Roll, I., McLaren, B. M., & Koedinger, K. R. (2016). Help helps, but only so much: Research on help seeking with intelligent tutoring systems. *International Journal of Artificial Intelligence in Education*, 26(1), 205–223.
- [5] Almeda, M. V. Q., Baker, R. S., & Corbett, A. (2017). Help Avoidance: When students should seek help, and the consequences of failing to do so. In *Meeting of the Cognitive Science Society* (Vol. 2428, p. 2433). Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science*, 13(4), 467–505. doi:10.1016/0364-0213(89)90021-9.
- [6] Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science*, 13(4), 467–505. doi:10.1016/0364-0213(89)90021-9.
- [7] Anderson, J. R. (1993). *Rules of the Mind*. Hillsdale: Lawrence Erlbaum Associates.
- [8] Arroyo, I., Beck, J., Woolf, B. P., Beal, C. R., & Schultz, K. (2000). Macro adapting animal watch to gender and cognitive differences with respect to hint interactivity and symbolism. In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), *Proceedings of the 5th International Conference on Intelligent Tutoring Systems, ITS 2000* (pp. 574–583). Berlin: Springer Verlag. doi:10.1007/3-540-45108-0_61.
- [9] Baker, R. S. J. d., Gowda, S. M., & Corbett, A. T. (2011). Towards predicting future transfer of learning. In G. Biswas, S. Bull, J. Kay, & A. Mitrovic (Eds.), *Lecture Notes in Computer Science: Artificial intelligence in Education: 15th International Conference, AIED 2011* (Vol. 6738, pp. 23–30). Berlin, Heidelberg: Springer. doi:10.1007/978-3-642-21869-9_6.
- [10] Bandura, A. (1982). Self-efficacy mechanism in human agency. *American Psychologist*, 37(2), 122.
- [11] Beck, J. E., Chang, K., Mostow, J., & Corbett, A. T. (2008). Does help help? Introducing the Bayesian evaluation and assessment methodology. In B. Woolf, E. Aimeur, R. Nkambou, & S. Lajoie (Eds.), *Proceedings of the 9th International Conference on Intelligent Tutoring Systems, ITS 2008* (pp. 383–394). Berlin: Springer.
- [12] Bong, M., Skaalvik, E. (2003). Academic self-concept and self-efficacy: How different are they really? *Educational Psych Review*. 15(1), pp. 1-40.
- [13] Brown, S. D., & Lent, R. W. (2006). Preparing adolescents to make career decisions: A social cognitive perspective. *Adolescence and Education*, 5, 201-223.
- [14] Childs, D. S. (2017). *Effects of Math Identity and Learning Opportunities on Racial Differences in Math Engagement, Advanced Course-Taking, and STEM Aspiration*. PhD Dissertation. Temple University.
- [15] Else-Quest, N. M., Mineo, C. C., & Higgins, A. (2013). Math and science attitudes and achievement at the intersection of gender and ethnicity. *Psychology of Women Quarterly*, 37(3), 293-309.
- [16] Goldin, I. M., Koedinger, K. R., & Aleven, V. (2012). Learner differences in hint processing. In K. Yacef, O. Zaïane, A. HersHKovitz, M. Yudelson, & J. Stamper (Eds.), *Proceedings of the 5th International Conference on Educational Data Mining (EDM 2012)* (pp. 73–80). Worcester: International Educational Data Mining Society.
- [17] Hogg, M. A. (2000). Subjective uncertainty reduction through self-categorization: A motivational theory of social identity processes. *European review of social psychology*, 11(1), 223-255.
- [18] Howley, I., Kanda, T., Hayashi, K., & Rosé, C. (2014). Effects of social presence and social role on helpseeking and learning. In G. Sagerer, M. Imai, T. Belpaeme, & A. Thomaz (Eds.), *HRI '14: Proceedings of the 2014 ACM/IEEE International Conference on Human-robot Interaction* (pp. 415–422). New York: ACM. doi:10.1145/2559636.2559667.
- [19] Hudley, A. H. C., & Mallinson, C. (2015). *Understanding English language variation in US schools*. Teachers College Press.
- [20] Karumbaiah, S., Ocumpaugh, J., Labrum, M., & Baker, R. S. (2019b). Temporally rich features capture variable performance associated with elementary students' lower math self-concept. In *Proceedings of the Workshop on Online Learning and social-Emotional Learning at the 9th International Conference on Learning Analytics and Knowledge (LAK)*.
- [21] Khachatryan, G. A., Romashov, A. V., Khachatryan, A. R., Gaudino, S. J., Khachatryan, J. M., Guarian, K. R., & Yufa, N. V. (2014). Reasoning Mind Genie 2: An intelligent tutoring system as a vehicle for international transfer of instructional methods in mathematics. *International Journal of Artificial Intelligence in Education*, 24(3), 333-382.
- [22] Kimble, G. A. (1987). The scientific value of undergraduate research participation. *American Psychologist*, 42(3), 267-268.
- [23] Klassen, R. M. (2004). Optimism and realism: A review of self-efficacy from a cross-cultural perspective. *International Journal of Psychology*, 39(3), 205-230.
- [24] Koedinger, K. R., & Aleven, V. (2007). Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, 19(3), 239-264.
- [25] Long, Y., & Aleven, V. (2013). Skill diaries: Improve student learning in an intelligent tutoring system with periodic self-assessment. In H. C. Lane, K. Yacef, J. Mostow, & P. Pavlik (Eds.), *Proceedings of the 16th International Conference on Artificial Intelligence in Education, AIED 2013* (pp. 249–258). Berlin Heidelberg: Springer. doi:10.1007/978-3-642-39112-5_26.
- [26] Mathews, M., Mitrović, T., & Thomson, D. (2008). Analysing high-level help-seeking behaviour in ITSs. In W. Nejdl, J. Kay, P. Pu, & E. Herder (Eds.), *Adaptive Hypermedia and Adaptive Web-based Systems: 5th*

- International Conference, AH 2008* (pp. 312–315). Berlin, Heidelberg: Springer. doi:10.1007/978-3-540-70987-9_42.
- [27] Marsh, H. W., Trautwein, U., Lüdtke, O., Köller, O., and Baumert, J. (2005). Academic self-concept, interest, grades, and standardized test scores: Reciprocal effects models of causal ordering. *Child Development*, 76(2):397–416.
- [28] McKendree, J. (1990). Effective feedback content for tutoring complex skills. *Human-Computer Interaction*, 5(4), 381–413. doi:10.1207/s15327051hci0504_2.
- [29] Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N., & Heffernan, C. (2014). Population validity for Educational Data Mining models: A case study in affect detection. *British Journal of Educational Technology*, 45(3), 487–501..
- [30] Ogan, A., Walker, E., Baker, R., Rodrigo, M. M. T., Soriano, J. C., & Castro, M. J. (2015). Towards understanding how to assess help-seeking behavior across cultures. *International Journal of Artificial Intelligence in Education*, 25(2), 229–248.
- [31] Pardos, Z. A., & Heffernan, N. T. (2010, June). Modeling individualization in a bayesian networks implementation of knowledge tracing. In *International Conference on User Modeling, Adaptation, and Personalization* (pp. 255–266). Springer, Berlin, Heidelberg.
- [32] Razzaq, L., & Heffernan, N. T. (2010). Hints: Is it better to give or wait to be asked? In V. Aleven, J. Kay, & J. Mostow (Eds.), *Lecture Notes in Computer Science: Proceedings of the 10th International Conference on Intelligent Tutoring Systems, ITS 2010* (Vol. 1, pp. 115–124). Berlin: Springer.
- [33] Roll, I., Baker, R. S. J. D., Aleven, V., & Koedinger, K. R. (2014). On the benefits of seeking (and avoiding) help in online problem-solving environments. *Journal of the Learning Sciences*, 23(4), 537–560. doi:10.1080/10508406.2014.883977.
- [34] Schunk, D. H., & Pajares, F. (2005). Competence perceptions and academic functioning. *Handbook of Competence and Motivation*, 85, 104.
- [35] Shih, B., Koedinger, K. R., & Scheines, R. (2008). A response time model for bottom-out hints as worked examples. In R. S. J. d. Baker, T. Barnes, & J. Beck (Eds.), *Proceedings of the 1st International Conference on Educational Data Mining, EDM 2008* (pp. 117–126). Montreal, Canada.
- [36] Stamper, J., Barnes, T., & Croy, M. (2011). Enhancing the automatic generation of hints with expert seeding. *International Journal of Artificial Intelligence in Education*, 21(1–2), 153–167. doi:10.3233/JAI-2011-021.
- [37] Steele, C. M. (1997). A threat in the air: How stereotypes shape intellectual identity and performance. *American Psychologist*, 52(6), 613.
- [38] Tai, M., Arroyo, I., & Woolf, B. (2013). Teammate relationships improve help-seeking behavior in an intelligent tutoring system. In H. C. Lane, K. Yacef, J. Mostow, & P. Pavlik (Eds.), *Lecture Notes in Computer Science: Artificial Intelligence in Education* (Vol. 7926, pp. 239–248). Berlin, Heidelberg: Springer. doi:10.1007/978-3-642-39112-5_25.
- [39] Tessler, R.C., and Schwartz, S.H. 1972. Help seeking, selfesteem, and achievement motivation: an attributional analysis. *Journal of Personality and Social Psychology* 21, 3 (1972), 318–326.
- [40] Texas Education Agency. (2018, October 4). *State and School District Summary*. Retrieved from http://www.texaseducationinfo.org/infopage/Summary_Report_Glossary.pdf
- [41] Texas Education Agency. (2018, July). *District Type Glossary of Terms*. Retrieved from <https://tea.texas.gov/acctres/analyze/1617/gloss1617.html#Major20Urban>
- [42] The Texas Tribune. (2018, August 24). *State and School District Summary*. Retrieved from <https://www.texastribune.org/2018/08/24/texas-school-districts-a-f-grades-takeaways/>
- [43] Urdan, T., & Pajares, F. (Eds.). (2006). *Self-efficacy beliefs of adolescents*. IAP.
- [44] Usher, E. L., & Pajares, F. (2006). Sources of academic and self-regulatory efficacy beliefs of entering middle school students. *Contemporary Educational Psychology*, 31(2), 125–141.
- [45] Vaessen, B. E., Prins, F. J., & Jeurig, J. (2014). University students' achievement goals and help-seeking strategies in an intelligent tutoring system. *Computers & Education*, 72, 196–208.
- [46] VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, 16(3), 227–265.
- [47] Wang, Y., & Beck, J. (2013, July). Class vs. student in a bayesian network student model. In *International Conference on Artificial Intelligence in Education* (pp. 151–160). Springer, Berlin, Heidelberg.
- [48] Wood, H., & Wood, D. (1999). Help seeking, learning and contingent tutoring. *Computers & Education*, 33(2/3), 153–169.
- [49] Yudelson, M., Fancsali, S., Ritter, S., Berman, S., Nixon, T., & Joshi, A. (2014, July). Better data beats big data. In *Educational Data Mining 2014*.
- [50] Zeldin, A. L., & Pajares, F. (2000). Against the odds: Self-efficacy beliefs of women in mathematical, scientific, and technological careers. *American Educational Research Journal*, 37(1), 215–246.
- [51] Zeldin, A. L., Britner, S. L., & Pajares, F. (2008). A comparative study of the self-efficacy beliefs of successful men and women in mathematics, science, and technology careers. *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching*, 45(9), 1036–1058.

Exploring Neural Network Models for the Classification of Students in Highly Interactive Environments

Tanja Käser
Graduate School of Education
Stanford University
tkaeser@stanford.edu

Daniel L. Schwartz
Graduate School of Education
Stanford University
daniel.schwartz@stanford.edu

ABSTRACT

Open-ended learning environments (OELEs) allow students to freely interact with the content and to discover important principles and concepts of the learning domain on their own. However, only some students possess the necessary skills for efficient and effective exploration. Guidance in the form of targeted interventions or feedback therefore has the potential to improve educational outcomes. A promising approach for adaptation in OELEs is the design of interventions based on the detection of characteristic learning behaviors through offline clustering, followed by a real-time classification of new students. In this paper, we explore the possibility of using recurrent neural network (RNN) models for this online classification task. We extensively evaluate the predictive performance of different variants of RNNs, namely long-short term memory models and gated recurrent units, and different architectures on a data set collected from an exploration-based educational game. We also compare the prediction accuracy of the different RNN models to the performance of traditional classifiers on the same data set. Our results demonstrate that RNNs perform similar or better than traditional methods regarding early classification and therefore constitute a promising alternative for the online classification of new students.

Keywords

recurrent neural networks, online classification, exploration environments, learning behavior

1. INTRODUCTION

Over the last years, there has been a rise in OELEs such as educational games or simulations. These environments allow students to freely interact with the content and (ideally) infer the concepts and principles of the learning domain through their exploration. However, previous research [20, 31, 24] has demonstrated that few students possess the problem solving and inquiry skills necessary to efficiently and effectively explore the space. Individualized guidance in the

form of adaptive interventions or feedback therefore have the potential to improve students' exploration strategies and at the same time optimize the educational outcomes.

Traditionally, adaptation in computer-based learning environments has been based on the predictions of the student model. A large body of work has focused on developing student models that are able to accurately represent student knowledge. One of the most popular student modeling approaches is Bayesian Knowledge Tracing (BKT) [8], a technique that has been constantly refined and improved over the years, e.g., [34, 35]. Other widely used approaches are based on item response theory, such as the Additive Factors Model [5, 6] and Performance Factors Analysis [28]. Furthermore, dynamic Bayesian networks, e.g., [13, 18] have been used to model student knowledge. All of these approaches are based on the assumption that the knowledge of the student can be represented through a set of skills (knowledge components) and that we can infer the knowledge about a specific skill based on students' answers to tasks associated with this skill. OELEs do not fulfill these criteria as they (usually) do not provide specific sequences of tasks or explicitly define knowledge components. Therefore, the introduced student modeling techniques cannot be (directly) applied to such environments.

A prominent idea in the literature is to provide adaptation based on detected (and analyzed) learning behaviors. This idea has been formalized into a user modeling framework [15]: First, offline clustering is used to identify different types of student behaviors. The adaptive components of the environment are then designed with respect to the different behaviors found. Second, an online classification algorithm assigns new students to one of the clusters (and the corresponding intervention) in real time. A large amount of previous research has focused on the offline clustering part of the framework, applying clustering approaches to identify different types of learners [25, 3, 10]. [12] have represented student activity patterns in massive open online courses (MOOCs) using behavior state-transition graphs and demonstrated that the extracted patterns can be interpreted. Other work assessed students' problem solving behaviors in a game-based learning environment [32]. The full framework has been successfully applied to an environment for learning common artificial intelligence algorithms [15]. Other researchers [16] have used the framework to predict the mathematical learning patterns of students. Recently, the framework has been used to build student models for a more complex simulation of electric circuits [11]. To summarize, the presented research has mostly focused on offline clustering or the appli-

Tanja Käser and Daniel L. Schwartz "Exploring Neural Network Models for the Classification of Students in Highly Interactive Environments" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 109 - 118

cation of the framework to different learning domains, performing online classification using standard algorithms such as k-nearest neighbor [2].

Recurrent neural networks (RNN) have been successfully used for a variety of sequence classification problems such as sentiment analysis, e.g., [23] and video, e.g., [9]. RNNs have also been used in the educational community, for example to model student knowledge [30], to provide personalized recommendations in MOOCs [27], or to improve sensor-free affect detection [4]. Furthermore, RNNs have also been employed for classification problems. [26] suggested the use of long-short term memory (LSTM) models to classify learner behavior from touchscreen data. Other work [1] used LSTMs for the classification of problem-solving behaviors.

In this paper, we explore the use of RNNs for online classification: we assume the offline clustering solution to be given and train different classifiers to predict the cluster label of a new student early on during interaction with the OELE. We hypothesize that the ability of RNNs to handle sequences of arbitrary length, allowing them to accumulate the relevant information over each time step, may benefit the online classification task. We investigate different types of RNNs, varying the models along three dimensions: the internal node structure used (LSTMs and gated recurrent units (GRU)), the depth of the network (number of layers), and the number of nodes in the hidden layers. We also train the models to either predict the whole sequence, i.e., outputting the cluster label at each time step, or only predict the cluster label at the end of the sequence. The former approach has the advantage that the model is able to predict the cluster label at any point in time. The latter approach requires training different models to make predictions at specific time points, but enables the models to optimize predictions for the given point in time. We extensively evaluate and compare the predictive performance of all the different RNN models on a data set collected from an OELE [17]. Our results demonstrate that RNNs trained to predict the cluster label at each time step reach a similar predictive performance in early classification as the RNNs trained to predict the cluster label at the end of the sequence. Furthermore, despite the smaller number of parameters, GRU models tend to achieve a classification accuracy similar to LSTM models. We also compare the RNN models to traditional classifiers on the same data set. Our findings show that the RNN models perform similarly or better than the traditional approaches regarding the prediction of cluster labels early in the game. Earlier prediction of cluster labels allows to provide targeted guidance sooner. We therefore conclude that the use of RNNs for the online classification of student types is promising.

2. DATASET

The data set at hand was collected from a short interactive game aiming at assessing students' exploration choices.

Training Environment. *TugLet* is an interactive game designed to assess students' exploration behavior. The topic of the game revolves around a tug-of-war. Players can choose between two modes (illustrated in Fig. 1): they can engage in inquiry by simulating tug-of-war set-ups (*Explore*) or they can try to predict the winning side of specific tug-of-war set-ups and receive right-wrong feedback (*Challenge*). The

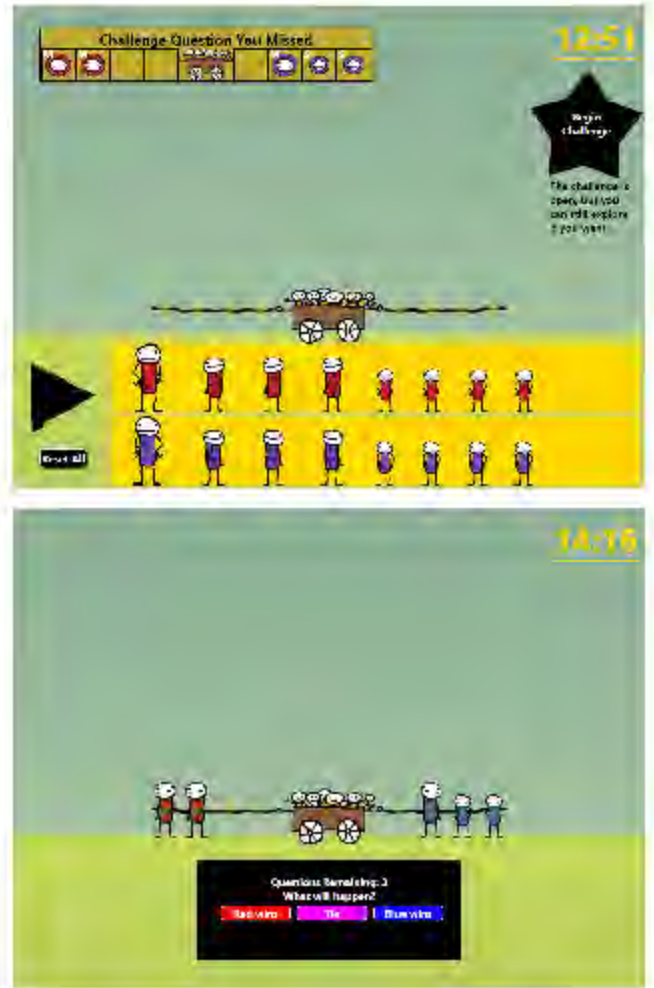


Figure 1: Snapshots of *Explore* (top) and *Challenge* modes (bottom). In *Explore* mode, children can set-up different tug-of-war teams and observe the outcome. In *Challenge* mode, children have to determine the winning side of specific tug-of-war set-ups.

Challenge mode consists of a maximum of eight problems ordered by increasing difficulty. The eight problems consist of one very easy question followed by two easy questions, two medium questions, and three difficult questions. If the student answers a problem incorrectly, (s)he is put back into *Explore* mode. The student is however free to choose to go back to the *Challenge* mode at any point in time. The game is over after players solve eight problems in a row correctly. The learning goal of the game, which is not revealed to the player, is to discover the mathematical principles underlying the tug-of-war.

Data Set. The data set consists of log files of 229 students attending the 8th grade of two different middle schools. The total number of observations in this data set is 10'258. One observation corresponds to either one set-up tested in *Explore* mode or one question answered in *Challenge* mode. The length l of the observation sequences varies between $l = 12$ and $l = 127$. All students in this data set managed to pass the game.

Clustering Solution. Previous work [17] has shown that the learning outcome (measured by an external posttest) is

not only influenced by students' exploration choices (*Explore* vs. *Challenge*) but also by the quality of their inquiry strategies. It was furthermore shown [19] that students can be grouped into six different clusters based on these detected strategies. These clusters can be semantically interpreted: cluster 1 captures students who systematically explore and try to understand the mathematical principles behind the tug-of-war. Cluster 3 consists of students who pass the game fast by only using *Challenge* mode. Students in cluster 6 also do not explore, but take a long time to pass the game. Students in cluster 4 on the other hand simulate many different tug-of-war configurations in *Explore* mode, without success. Cluster 2 lies in-between clusters 1 and 3, and exploration behaviors in cluster 5 are a mix between those in cluster 3 and cluster 6. The clusters are not only correlated to performance in an external posttest, but also predict academic achievement more broadly [19].

The features serving as an input for the clustering are extracted by level: children need to answer eight *Challenge* questions in a row correctly to pass the game and therefore the game can be divided into eight levels. Level n is reached the first time the student answers exactly n questions in a row correctly. The features used for clustering consist of the following cumulative counts extracted for level $n \in [1, 8]$: the number of *Challenge* problems NC_n needed to reach level n , the total number of tug-of-war set-ups NE_n simulated in *Explore* mode before reaching level n , and the number of tug-of-war set-ups NSE_n simulated in *Explore* mode which are classified as reflecting systematic inquiry (see [17] for a definition of systematic inquiry) until reaching level n . Therefore, the input features used for the clustering are $\mathbf{NC} = [NC_1, \dots, NC_8]$, $\mathbf{NE} = [NE_1, \dots, NE_8]$, and $\mathbf{NSE} = [NSE_1, \dots, NSE_8]$. The cluster solution is then found by computing the pair-wise dissimilarities between all students for each feature using the Euclidean distance as a similarity measure and subsequently performing a pair-wise clustering [14]. The optimal number of clusters is determined by the Bayesian Information Criterion (BIC) [29]. In the following, we will use the presented clustering solution as ground truth for our classification task.

3. ONLINE CLASSIFICATION OF NEW STUDENTS

Ideally, the output of the clustering algorithm enables us to characterize different student behaviors and to design interventions or feedback based on the detected behaviors. Ideally, we are also able to characterize the performance of new learners in real time in order to deliver a targeted intervention as soon as possible. The corresponding framework is illustrated in Fig. 2. In the case of our data set, students assigned to cluster 4 might for example get hints on how to explore systematically, while students from cluster 3 will be prompted to use *Explore* mode in order to figure out the principles governing the tug-of-war.

RNN models are a family of neural network models able to handle sequences of arbitrary lengths. They are especially suited for time-series data and are able to represent the relevant information over a sequence of time steps. We therefore adapt different types of RNNs for the online classification task (marked in dark blue in Fig 2). RNNs map a sequence of input features $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T$ to a sequence of output features $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T$. They maintain a sequence of hidden states $\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_T$. These hidden states capture

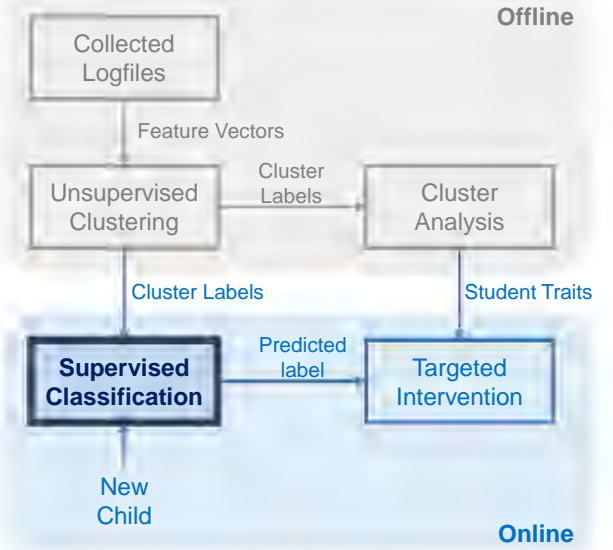


Figure 2: Students are clustered offline and targeted interventions are designed based on the (semantic) cluster interpretation. New students are then classified online. In this paper, we focus on the online classification task (marked in dark blue).

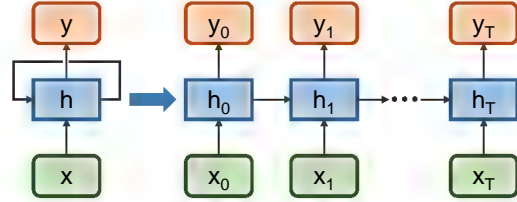


Figure 3: Simple RNN unrolled over T time steps. \mathbf{x} denotes the input feature vectors, \mathbf{y} denotes the output feature vectors and the hidden states are represented by \mathbf{h} .

relevant information from past observations which will influence future predictions. Figure 3 shows an illustration of a simple RNN model.

3.1 Long-Short Term Memory Classification

Long-short term memory (LSTM) models are a powerful modification of the RNN architecture.

Specification. LSTM models replace each hidden state \mathbf{h}_t by an LSTM cell unit with additional gating parameters. These parameters determine when to forget or retain previous information. The update equations of an LSTM are as follows:

$$\mathbf{f}_t = \sigma(W_{fx}\mathbf{x}_t + W_{fh}\mathbf{h}_{t-1} + \mathbf{b}_f) \quad (1)$$

$$\mathbf{i}_t = \sigma(W_{ix}\mathbf{x}_t + W_{ih}\mathbf{h}_{t-1} + \mathbf{b}_i) \quad (2)$$

$$C'_t = \tanh(W_{Cx}\mathbf{x}_t + W_{Ch}\mathbf{h}_{t-1} + \mathbf{b}_C) \quad (3)$$

$$C_t = \mathbf{f}_t \times C_{t-1} + \mathbf{i}_t \times C'_t \quad (4)$$

$$\mathbf{o}_t = \sigma(W_{ox}\mathbf{x}_t + W_{oh}\mathbf{h}_{t-1} + \mathbf{b}_o) \quad (5)$$

$$\mathbf{h}_t = \mathbf{o}_t \times \tanh(C_t) \quad (6)$$

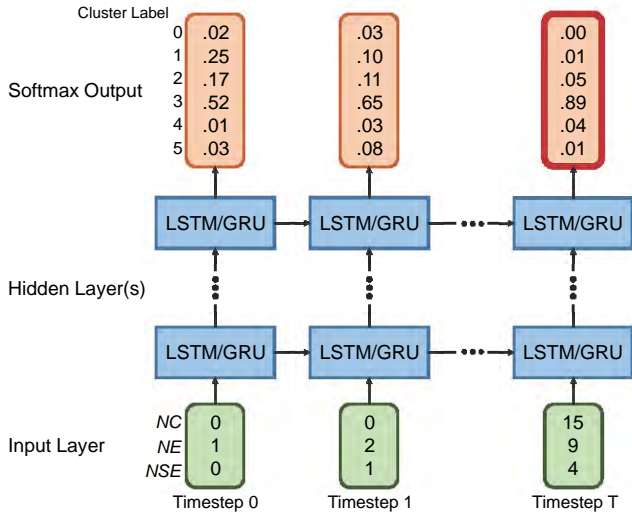


Figure 4: RNN model (LSTM or GRU units) with n hidden layers for an example student over T time steps. The input vector \mathbf{x}_t contains the counts NC_t , NE_t , and NSE_t at each time step t . The output vector $\hat{\mathbf{y}}_t$ can be interpreted as a probability distribution over the different cluster labels.

Here, \mathbf{f}_t , \mathbf{i}_t , and \mathbf{o}_t represent the forget, input, and output gates of the LSTM cell unit C_t . C'_t denotes an intermediate candidate cell state. The different weight matrices are described by W and the \mathbf{b} stands for bias.

Modeling. For our task, the input vector \mathbf{x}_t encodes the clustering features at each time step t . We input the counts for each time step t , i.e., $\mathbf{x}_t = [NC_t, NE_t, NSE_t]$. Let us assume that a hypothetical student m tested three different set-ups, the first two being random trials and the third one being systematic, followed by answering two *Challenge* questions. The input features for this student over the five described time steps are as follows: $\mathbf{x}_{m,1} = [0, 1, 0]$, $\mathbf{x}_{m,2} = [0, 2, 0]$, $\mathbf{x}_{m,3} = [0, 3, 1]$, $\mathbf{x}_{m,4} = [1, 3, 1]$, $\mathbf{x}_{m,5} = [2, 3, 1]$. Figure 4 details T time steps of an example student.

The output vectors $\hat{\mathbf{y}}_t$ represent the (predicted) cluster labels of the students: the output layer of the model uses the softmax function to normalize the vectors to sum to 1 such that the values within these output vectors can be thought of as probabilities for the different cluster labels (see Fig. 4). When training the model, we provide the cluster labels found during clustering as ground truth. Note that we use a one-hot encoding of the cluster labels, i.e. for a student m belonging to cluster $k = 2$, $\mathbf{y}_{m,t} = [0, 0, 1, 0, 0, 0]$. The chosen model predicts the cluster label of the student at each time step t , augmenting the amount of available data and increasing flexibility, as it allows to predict the cluster label of a specific child at any point in time. We denote this type of model as $LSTM_{Seq}$.

Note that $\mathbf{y}_{m,1} = \mathbf{y}_{m,2} = \dots = \mathbf{y}_{m,T}$ for all students m , because during clustering each student is assigned a fixed cluster label based on the whole sequence. Given that the cluster labels are fixed, we can also design the model to only output the cluster label at the end, i.e. train the LSTM to predict the cluster label at the end of a given input sequence. Instead of computing the training loss over the whole sequence, we calculate the loss only for the last output $\hat{\mathbf{y}}_T$ of the sequence (marked with red in Fig. 4). When using this type of model, we have to train a separate model for each

prediction time point. In our case, we will train separate models for each level. We call this model $LSTM_{End}$.

Stacking multiple LSTM layers (see hidden layers in Fig. 4) is another possible variation of the architecture, i.e., the vector \mathbf{h}_t of layer $n - 1$ serves as an input for layer n . Stacking layers adds levels of abstraction of input observations over time, for example enabling representation of the problem at different time scales. We will denote LSTM models with n hidden layers, where $n > 1$, as $LSTM_{Seq,n}$ or $LSTM_{End,n}$.

3.2 Gated Recurrent Unit Classification

Gated recurrent unit (GRU) models are another powerful modification of RNN models. In contrast to LSTMs, they are less complex, making training more efficient.

Specification. Similar to LSTM models, GRU models replace each hidden unit \mathbf{h}_t by a GRU cell unit with additional gating parameters. GRUs use update and reset gates, deciding what information should be passed to the output. The update equations of a GRU are as follows:

$$\mathbf{z}_t = \sigma(W_{fz}\mathbf{x}_t + W_{fh}\mathbf{h}_{t-1} + \mathbf{b}_z) \quad (7)$$

$$\mathbf{r}_t = \sigma(W_{ix}\mathbf{x}_t + W_{ih}\mathbf{h}_{t-1} + \mathbf{b}_i) \quad (8)$$

$$\mathbf{h}'_t = \tanh(W_{h'x}\mathbf{x}_t + r_t \times W_{h'h}\mathbf{h}_{t-1} + \mathbf{b}'_h) \quad (9)$$

$$\mathbf{h}_t = \mathbf{z}_t \times \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \times \mathbf{h}'_t \quad (10)$$

\mathbf{z}_t and \mathbf{r}_t represent the update and reset gate of the GRU cell unit. \mathbf{h}'_t denotes an intermediate candidate hidden state. The different weight matrices are described by W and the \mathbf{b} stands for bias.

Modeling. Just as for the LSTM models, the clustering features at each time step t are represented by the input vector \mathbf{x}_t , i.e., $\mathbf{x}_t = [NC_t, NE_t, NSE_t]$. The input sequence of an example student is given in Fig. 4. We also use the exact same description for the output layer of the GRU: the output layer of the model uses the softmax function to normalize the vectors to sum to 1 such that the values within the output vectors $\hat{\mathbf{y}}_t$ can be interpreted as probabilities for the different cluster labels (see Fig. 4). We again train a model on the whole output sequences of the students able to predict the cluster label at each time step t . We denote this model with GRU_{Seq} . We also train one GRU model per level, where we compute the loss of the model only for the last output $\hat{\mathbf{y}}_T$ of the sequence (marked with red in Fig. 4). We denote this model with GRU_{End} . Finally, similar to LSTM models, we can also stack GRU models on top of each other. We will call models with n hidden layers, where $n > 1$, $GRU_{Seq,n}$ or $GRU_{End,n}$.

4. EXPERIMENTAL EVALUATION

We evaluated the predictive accuracy of the variations of RNN models on the data set described in Section 2. We also compared them to the following popular traditional classifiers using the same data set: k-Nearest-Neighbor (kNN) and random forests (RF). While these classifiers are well tested and efficient to train, they require features being the same length for each student and therefore need to be trained for fixed points in time. RNNs on the other hand represent the relevant information over time, possibly enabling a more accurate classification of students early in the game.

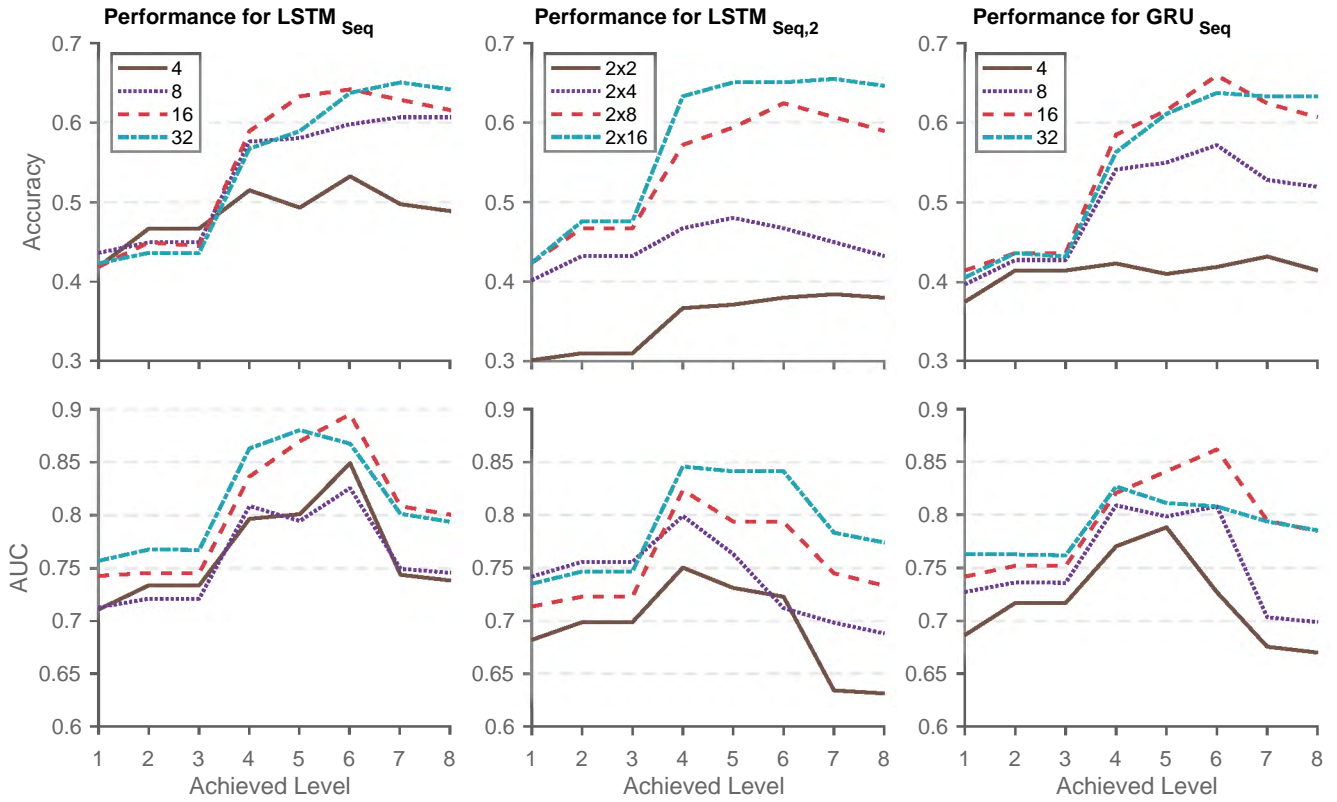


Figure 5: Accuracy (top) and AUC (bottom) of the $LSTM_{Seq}$, $LSTM_{Seq,2}$, and GRU_{Seq} models by achieved level. Both measures increase up to level 6 and stagnate or deteriorate afterward. This is probably due to the fact that the models are trained to predict the cluster label after each time step t , i.e. training loss is optimized over the whole sequence of observations.

Experimental Setup. We applied a train-test setting, i.e. parameters were fit on the training data set and performance of the methods was evaluated on the test data set. Predictive performance was evaluated using the accuracy as well as the micro-averaged area under the ROC curve (AUC). The accuracy is a measure that can be interpreted easily. The cluster solutions for both data sets are not balanced. We used the AUC as an additional performance measure as it is robust to class imbalance.

For all methods, we used a student-stratified (i.e. dividing the folds by students) 10-fold cross validation. Within each fold f , we re-clustered the students of the training data set of f to obtain the output features \mathbf{y} , i.e. the cluster labels, for training. We purposely did not use the original cluster labels from the solution found on the whole data set (including training and test data) for training, to prevent dependencies to the cluster labels of the test data set. The average cluster stability [22] between the 10 different training data sets and the original cluster solution was 0.87, i.e., on average 87% of the samples received the same label on the training data set as on the original data set. Therefore, 0.87 constitutes an upper bound for the accuracy of the classifiers.

All the RNN models were implemented using Keras [7] with Theano [33] as backend. Categorical crossentropy was used to calculate loss and ADAM was used as the optimizer. The models were trained for $e = 100$ epochs. For all types of RNN models, we used post-padding and masking to account for the different sequence lengths.

For the traditional classifiers, we trained one model for each

level n of the game. The input vector \mathbf{x}_n of each model therefore encodes the clustering features exactly at level n , i.e., $\mathbf{x}_n = [NC_n, NE_n, NSE_n]$.

We determined the optimal number k_o of nearest neighbors for the kNN classifier as follows: within each fold f , we randomly put 10% of the students from the training data set in a validation data set and selected the number of nearest neighbors $k_{o,f}$ yielding the best performance in terms of accuracy on this validation data set. We then predicted the cluster labels of the test data set using the labels of the $k_{o,f}$ nearest neighbors.

For the RF method, we trained $B = 100$ binary decision trees using bootstrapping with re-sampling ($r_f = 1.5 \cdot M_f$, with M_f being the number of samples in the training data set of fold f).

RNN Models returning a sequence of outputs. We varied the parameters of our RNN models outputting the whole sequence of cluster labels along three dimensions: the structure of the hidden layer(s) (LSTM or GRU), the number of hidden layers, and the dimension of the hidden state within a hidden layer. Specifically, we computed the predictive performance for the following models: a 1-layer LSTM ($LSTM_{Seq}$) with a d_h -dimensional hidden state where $d_h \in \{4, 8, 16, 32\}$, a 2-layer LSTM ($LSTM_{Seq,2}$) with a d_h -dimensional hidden state per layer where $d_h \in \{2, 4, 8, 16\}$, and a 1-layer GRU (GRU_{Seq}) with a d_h -dimensional hidden state where $d_h \in \{4, 8, 16, 32\}$. Figure 5 illustrates the predictive performance in terms of accuracy and AUC for the three

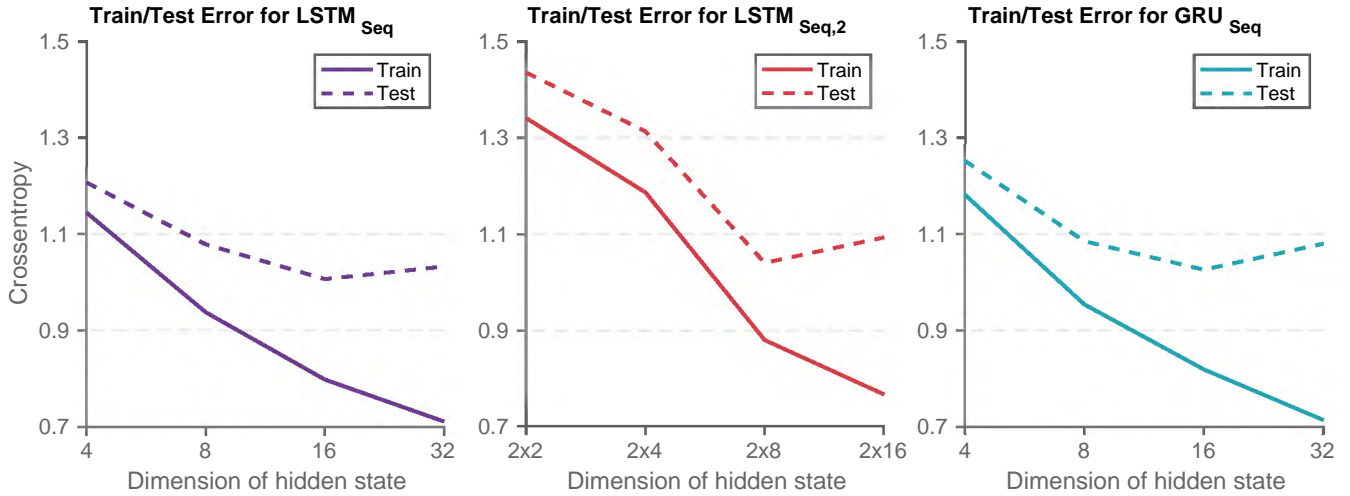


Figure 6: Categorical crossentropy of the LSTM_{Seq}, LSTM_{Seq,2}, and GRU_{Seq} models by the number of dimensions of the hidden state. The average test error begins to increase when using more than $d_h = 16$ hidden dimensions, while the training error still decreases.

different architectures by achieved level.

The LSTM_{Seq} models reach a poor accuracy when $d_h = 4$. There is also not much difference between the accuracy at level 1 (0.42) and at level 8 (0.49). The accuracy improves substantially when increasing the dimension of the hidden state to $d_h = 8$, $d_h = 16$, or $d_h = 32$. We especially observe a jump in accuracy between levels 3 (e.g., Accuracy₈ = 0.45) and 4 (e.g., Accuracy₈ = 0.58). For $d_h = 32$, there is a second jump between levels 5 (Accuracy₃₂ = 0.59) and 6 (Accuracy₃₂ = 0.64). These jumps in accuracy correspond to jumps in the difficulty of the *Challenge* questions: the eight problems consist of one very easy question followed by two easy questions, two medium questions, and three difficult questions. There is no increase in accuracy between levels 1 and 3 as most students passed these easy levels very quickly. We observe a similar picture for the AUC. The AUC increases with the number of dimensions d_h of the hidden state. Note that the AUC again jumps between levels 3 (e.g., AUC₈ = 0.72) and 4 (e.g., AUC₈ = 0.81).

For the LSTM_{Seq,2} models, predictive performance again increases with the number of dimensions of the hidden state within a layer. For this type of models, using a 2-dimensional hidden state or a 4-dimensional hidden state per layer leads to a low accuracy. Increasing to an 8-dimensional hidden state or a 16-dimensional hidden state per layer yields a large improvement in accuracy and these models also show a jump in accuracy between level 3 (e.g., Accuracy_{2x16} = 0.48) and level 4 (e.g., Accuracy_{2x16} = 0.63). The AUC also increases with an increasing number of dimensions d_h per hidden layer. The LSTM_{Seq,2} models' accuracy is in range with the accuracy of the LSTM_{Seq} models for $d_h = 16/d_h = 2x8$ hidden dimensions as well as for $d_h = 32/d_h = 2x16$ hidden dimensions. However, the LSTM_{Seq} models show a higher AUC than the LSTM_{Seq,2} models, e.g., for $d_h = 16/d_h = 2x8$ hidden dimensions (for example at level 5: AUC_{LSTM_{Seq}} = 0.87, AUC_{LSTM_{Seq,2}} = 0.79).

Performance of the GRU_{Seq} models in terms of accuracy shows the same trends over time as for the LSTM_{Seq} mod-

els. Again, employing a 4-dimensional hidden state results in a low accuracy. When increasing the number of dimensions of the hidden state to $d_h = 8$, $d_h = 16$, or $d_h = 32$, the models are able to capture the jump in difficulty between level 3 (e.g., Accuracy₈ = 0.43) and level 4 (e.g., Accuracy₈ = 0.54). The architecture employing a 16-dimensional hidden state also shows a jump in accuracy between level 4 (Accuracy₁₆ = 0.59) and level 6 (Accuracy₁₆ = 0.66). The AUC again increases with an increasing number of dimensions of the hidden state. Applying $d_h = 32$ instead of $d_h = 16$ hidden dimensions does generally not increase the AUC, and is even worse for some levels, e.g., for level 6 (AUC₁₆ = 0.86, AUC₃₂ = 0.81). Generally performance is in range with the performance of the LSTM_{Seq} models in terms of accuracy for $d_h = 16$ and $d_h = 32$ hidden dimensions. Again, the AUC for the LSTM_{Seq} models tends to be higher than the AUC for the GRU_{Seq} models, for example for $d_h = 16$ hidden dimensions at the peak level 6 (AUC_{LSTM_{Seq}} = 0.90, AUC_{LSTM_{GRU}} = 0.86).

The performance increase of all models with a higher number of hidden dimensions is as expected. However, the danger of overfitting increases with a higher number of parameters. While our training and evaluation methods have measures for overfitting, such as the crossvalidation, in place, we investigated the relation between the average training and test error of the different configurations and the number of dimensions d_h of the hidden state. Figure 6 illustrates the average categorical crossentropy on the training data sets and test data sets. We observe that the difference between training error and test error starts to get bigger with an increased number of hidden dimensions. Specifically, there is a kink in the test error at $d_h = 16/d_h = 2x8$ hidden dimensions for all models. We therefore conclude that $d_h = 16/d_h = 2x8$ is the maximum number of hidden dimensions that should be used for this data set.

RNN Models returning the last output in the sequence. We also trained a range of RNN models returning

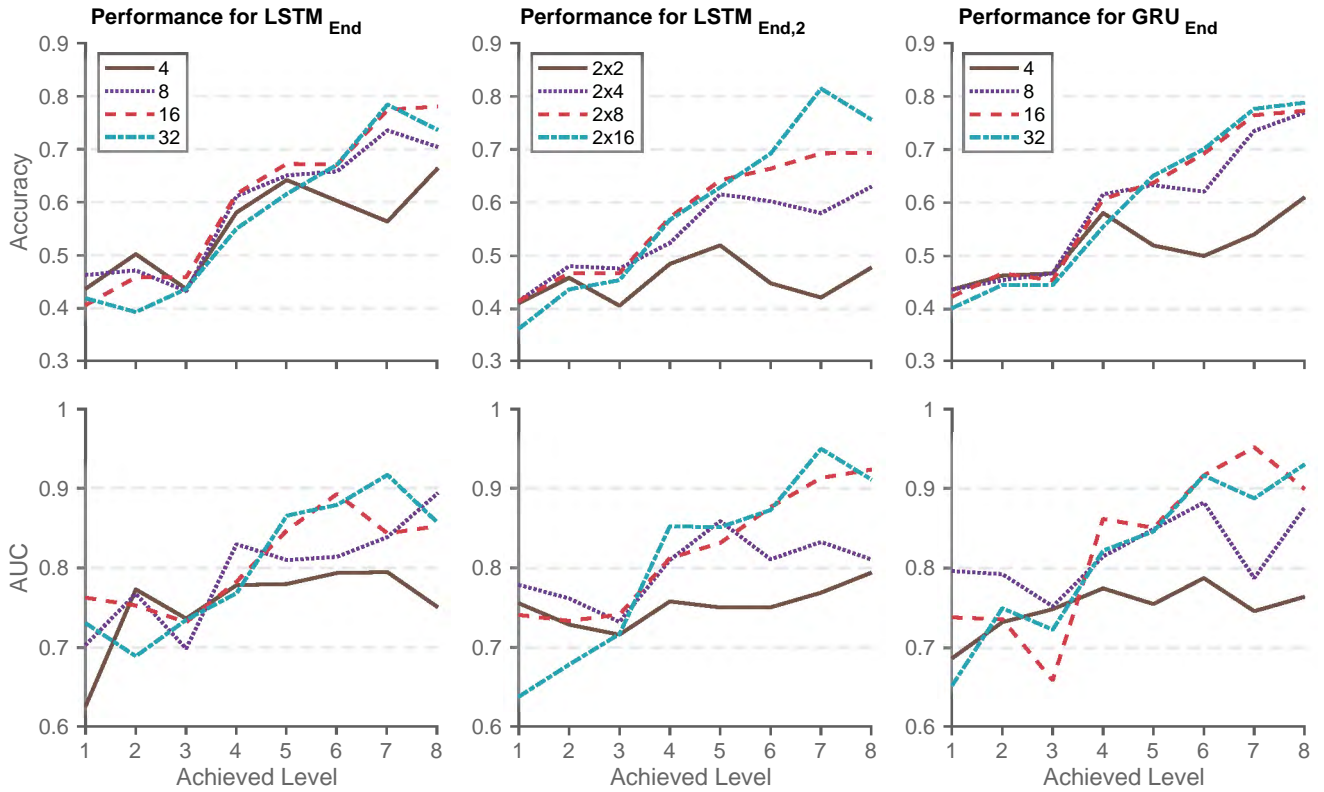


Figure 7: Accuracy (top) and AUC (bottom) of the LSTM_{End} , $\text{LSTM}_{\text{End},2}$, and GRU_{End} models by achieved level. Both measures increase over time and all models achieve a similar predictive performance for the higher numbers of hidden dimensions.

only the last output in the sequence, i.e. predicting the cluster label at the end of a given sequence. We varied the parameters of these models along the same dimensions as for the models predicting the whole sequence. Specifically, we computed the predictive performance for the following models: a 1-layer LSTM (LSTM_{End}) with a d_h -dimensional hidden state where $d_h \in \{4, 8, 16, 32\}$, a 2-layer LSTM ($\text{LSTM}_{\text{End},2}$) with a d_h -dimensional hidden state per layer where $d_h \in \{2, 4, 8, 16\}$, and a 1-layer GRU (GRU_{End}) with a d_h -dimensional hidden state where $d_h \in \{4, 8, 16, 32\}$. We trained one model for each level of the game. The predictive performance of all the models in terms of accuracy and AUC is illustrated in Fig. 7.

Similar to the models predicting sequences, the LSTM_{End} model achieves the lowest accuracy with $d_h = 4$. However, up to level 6, there is no big difference in accuracy between this model and the models with $d_h \geq 8$ hidden dimensions. All the LSTM_{End} models capture the jump in difficulty between level 3 and level 4. The models with higher-dimensional hidden states also capture the second jump happening after level 5. For $d_h = 16$, we for example see a jump in accuracy after level 3 (level 3: Accuracy = 0.46, level 4: Accuracy = 0.62) and a second jump after level 6 (level 6: Accuracy = 0.67, level 7: Accuracy = 0.77). Regarding the AUC, we observe superior performance of the models with $d_h = 16$ and $d_h = 32$ hidden dimensions after level 4. For these models, the AUC constantly increases until level 6 ($\text{AUC}_{16} = 0.89$, $\text{AUC}_{32} = 0.88$). As seen before, we do not observe any improvement in performance after level 6.

For the $\text{LSTM}_{\text{End},2}$ model, employing $d_h = 2$ hidden dimen-

sions per layer leads to the lowest achieved accuracy and there is also not much improvement over time. The models with $d_h > 2$ capture the increased difficulty after level 3 (e.g., level 3: $\text{Accuracy}_{2 \times 8} = 0.47$, level 4: $\text{Accuracy}_{2 \times 8} = 0.57$). Only the models using $d_h = 8$ or $d_h = 16$ hidden dimensions per layer manage to capture the second jump in difficulty (e.g., level 6: $\text{Accuracy}_{2 \times 16} = 0.69$, level 7: $\text{Accuracy}_{2 \times 16} = 0.81$). We observe a similar picture for the AUC: when using $d_h = 8$ or $d_h = 16$ hidden dimensions, there is a strong increase in AUC after level 3 (e.g., level 3: $\text{AUC}_{2 \times 8} = 0.74$, level 4: $\text{AUC}_{2 \times 8} = 0.81$) and after level 6 (e.g., level 6: $\text{AUC}_{2 \times 16} = 0.87$, level 7: $\text{AUC}_{2 \times 16} = 0.95$).

The accuracy plot of the GRU_{End} models looks similar to the accuracy plot of the LSTM_{End} models. Up to level 4, all models perform similarly (at level 4: $\text{Accuracy}_4 = 0.58$, $\text{Accuracy}_8 = 0.62$, $\text{Accuracy}_{16} = 0.61$, $\text{Accuracy}_{32} = 0.55$). For the higher levels, the model with $d_h = 4$ hidden dimensions shows the lowest accuracy. Using a model with an 8-dimensional hidden state nicely captures the jumps in accuracy between level 3 (Accuracy = 0.47) and level 4 (Accuracy = 0.62) and between level 6 (Accuracy = 0.62) and level 7 (Accuracy = 0.73). Increasing the number of hidden dimensions to $d_h = 16$ improves performance only from level 5 on (e.g., at level 6: $\text{Accuracy}_8 = 0.62$, $\text{Accuracy}_{16} = 0.69$). This model also captures the two jumps in accuracy. Using $d_h = 32$ hidden dimensions does not lead to any further improvements. The model employing a 4-dimensional hidden state performs worst for the AUC, with exception of level 3. When using $d_h = 16$ or $d_h = 32$ hidden dimen-

sions, the AUC again models the two jumps in difficulty (e.g., level 3: $AUC_{32} = 0.72$, level 4: $AUC_{32} = 0.82$, level 5: $AUC_{32} = 0.85$, level 6: $AUC_{32} = 0.92$). It also seems that using a higher number of hidden dimensions, i.e. $d_h > 8$, increases the stability of the AUC. With exception of the drop at level 3, the AUC of the model with $d_h = 16$ hidden dimensions nicely increases over time.

We again tested for overfitting, by comparing the average training and test loss of the different models. Just as for the models trained to predict the cluster label at each time step, we found that there is a kink at $d_h = 16$ hidden dimensions: while the training error still decreases for a higher number of d_h , the error on the test set increases. We therefore conclude that $d_h = 16$ is the maximum number of hidden dimensions that can be used.

‘Sequence versus End’. When comparing the RNN models trained for sequence prediction to the models trained to predict only the last output of the sequence, we observe that the performance plots show the same overall trends (see Fig. 5 and Fig. 7). For both types, predictive performance in terms of the accuracy is similar for the 1-layer LSTM and the 1-layer GRU models. The GRU_{Seq} model generally has a lower accuracy than the $LSTM_{Seq}$ model when using $d_h < 16$. The GRU_{End} model exhibits a lower accuracy than the $LSTM_{End}$ model only for $d_h = 4$. The models with two stacked layers, i.e. the 2-layer LSTM models, generally show a lower accuracy when employing a low number of hidden dimensions per layer (2×2 or 2×4).

For both the ‘sequence’ and the ‘end’ RNN models, all three model types achieve similar accuracies for $d_h = 8$ or $d_h = 16$ hidden dimensions. All RNN models show no improvement or even a drop in AUC after level 6. This effect is more pronounced for the models which are trained on the sequence, as their loss is optimized over the whole sequence. We further hypothesize that the length of the sequences at the higher levels might be too long for the RNN models to capture the relevant information, because even with the LSTM architecture, RNNs tend to struggle with very long data sequences. The main difference between the ‘sequence’ and the ‘end’ model is the larger increase of the accuracy with increasing levels. For example, for the $LSTM_{Seq}$ with $d_h = 16$ the accuracy at level 1 is 0.42 and the accuracy at level 7 is 0.63, while the accuracy of the $LSTM_{End}$ model with $d_h = 16$ is 0.41 at level 1 and 0.77 at level 7. Because the ‘end’ models are optimized to predict the last output of a sequence, they reach a higher accuracy at the end of the game (e.g. at level 8: $Accuracy_{GRU_{Seq},16} = 0.61$, $Accuracy_{GRU_{End},16} = 0.77$). The ‘sequence’ RNN models are optimal on average and therefore exhibit less variance over time and smoother accuracy and AUC curves. For the medium levels of the game, ‘sequence’ and ‘end’ RNN models perform similar in terms of accuracy and AUC.

All configurations exhibit a satisfying accuracy and a medium-high AUC for $d_h \geq 8$. As a comparison, a random classifier would achieve an accuracy of 0.17 and the accuracy of a classifier always predicting the majority label would be 0.32. The AUC of these two classifiers would be 0.5.

Comparison to traditional classifiers. We compared the predictive performance of selected RNN models to the predictive performance of the traditional classifiers. Because the RNN models show an increased performance with a

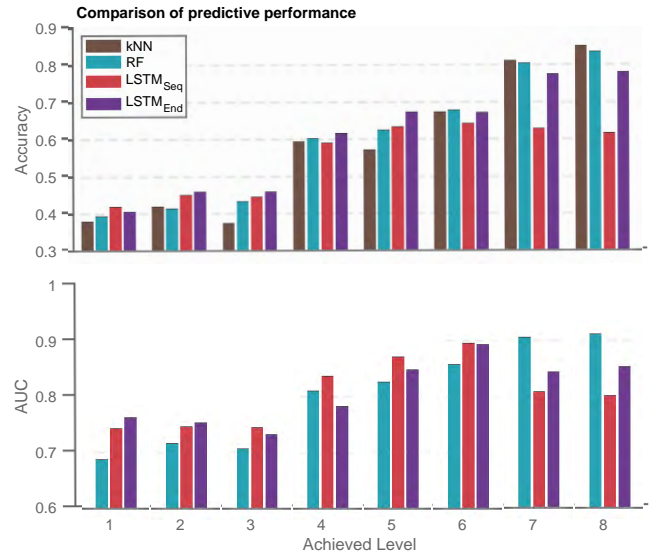


Figure 8: Accuracy (top) and AUC (bottom) of kNN, RF, $LSTM_{Seq}$, and $LSTM_{End}$ by achieved level. The RNNs achieve similar or better performance than the traditional classifiers up to level 6.

higher number of hidden dimensions, we used RNN models with the maximum number of $d_h = 16$ not resulting in overfitting for comparison. In case of the ‘sequence’ models, the $LSTM_{Seq}$ model achieved a similar accuracy, but a higher AUC than the two other model types for $d_h = 16$. We therefore selected the $LSTM_{Seq}$ model with $d_h = 16$ for comparison. In case of the ‘end’ models, the 1-layer LSTM and 1-layer GRU models achieved similar results, however, the LSTM models were better at predicting the jumps in difficulty. We therefore selected the $LSTM_{End}$ model with $d_h = 16$ for comparison. Figure 8 displays the accuracy and the AUC of the kNN classifier, the RF method, a 1-layer LSTM model for sequence predicting with $d_h = 16$ hidden dimensions ($LSTM_{Seq}$), and a 1-layer LSTM model for predicting the last output of the sequence with $d_h = 16$ hidden dimensions ($LSTM_{End}$).

We observe that the RNN models outperform the traditional classifiers for the first three levels regarding the accuracy (e.g., at level 3: $Accuracy_{kNN} = 0.38$, $Accuracy_{RF} = 0.43$, $Accuracy_{LSTM_{Seq}} = 0.45$, $Accuracy_{LSTM_{End}} = 0.46$). The same holds for the middle of the game, i.e. levels 4 – 6 (e.g., at level 5: $Accuracy_{kNN} = 0.57$, $Accuracy_{RF} = 0.63$, $Accuracy_{LSTM_{Seq}} = 0.63$, $Accuracy_{LSTM_{End}} = 0.67$). At the end of the game, the accuracy of the traditional classifiers is close to the stability of the clustering ($Accuracy_{kNN} = 0.85$, $Accuracy_{RF} = 0.83$).

For the RF approach and the two RNN models, we also computed the AUC (see Fig. 8 (bottom)). In contrast to the other three methods outputting probabilities for each cluster label, kNN just outputs the predicted cluster label and we therefore did not compute the AUC for this method. We observe that the $LSTM_{Seq}$ and the $LSTM_{End}$ models clearly outperform the RF method at the beginning of the game (e.g., at level 3: $AUC_{RF} = 0.71$, $AUC_{LSTM_{Seq}} = 0.75$, $AUC_{LSTM_{End}} = 0.73$). Also in the middle of the game between levels 4 and 6, the RNN models show a higher AUC than the RF method (e.g., at level 5: $AUC_{RF} = 0.82$,

$AUC_{LSTM_{Seq}} = 0.89$, $AUC_{LSTM_{End}} = 0.87$) with the exception of level 4. When looking at the whole sequence, we observe a similar picture as for the accuracy: the AUC of the RF method is clearly higher than the AUC of the RNN models (at level 8: $AUC_{RF} = 0.91$, $AUC_{LSTM_{Seq}} = 0.80$, $AUC_{LSTM_{End}} = 0.85$).

As already mentioned before we assume that the worse performance of the RNN models at the end of the game (level 7 and level 8) is due to the fact that the input sequences for the RNN models become too long. Note that while most students manage to reach level 5 within a reasonable time frame, the lengths of the complete sequences vary significantly between the students. The lower accuracy and AUC of the RNN models at the end of the game are not an issue in our case because we are interested in accurate predictions early in the game. While the RNN models show only a slightly increased accuracy in comparison to the traditional methods at the beginning and in the middle of the game, they consistently achieve a higher AUC up to level 6, demonstrating their robustness towards class imbalance.

5. DISCUSSION

OELs constitute a promising approach for learning. Ideally, the students learn the concepts and principles of a domain more deeply through exploration than if they are simply taught the principles and practice applying them. However, it has been shown that only a part of the students are able to effectively explore the space [20, 31, 24, 17]. Providing guidance to struggling students is therefore essential for educational success.

Because OELs allow the user to freely interact with the content, traditional student modeling approaches cannot directly be applied to provide adaptation to the student. Adaptation based on detected student behavior and strategies is therefore a promising approach. Previous work has used offline clustering to detect different student types, followed by online classification of new students [15, 16, 11].

In this paper, we focused on the task of online classification, i.e., predicting the student type (or behavior) early on during interaction with the environment to provide targeted guidance as early as possible. In contrast to previous work applying standard classifiers such as k-nearest-neighbor [15, 16, 11], we suggested the use of RNN models for the online classification task. While previous research has investigated the use of RNN models to classify the students according to their problem-solving behavior based on their whole sequence of interactions [1], to classify changing learner behaviors over time [26], or to detect affective states over time [4], we explored the possibility of using RNN models to predict a student's cluster label (fixed over time) as early as possible. We have extensively evaluated a variety of RNN models and compared their predictive performance to the performance of k-nearest neighbor and random forest classifiers. We have used the different levels of the game as specific time points for evaluation as they pose realistic time points for interventions. We have trained RNN models to predict the cluster label at each time step as well as RNN models optimized for predicting the cluster label at each level of the game.

Not unexpectedly, the RNN models trained per level as well as the traditional classifiers outperform the models trained for predicting the whole sequence at the higher levels (level 7 and 8) of the game. This is due to the averaging effect of the

performance of the 'sequence' RNN models: during training, the loss is computed for each time step of the sequence. Nevertheless, for level 4 and 5, which provide promising time points of intervention both in terms of accuracy of the different models as well as in terms of timing of intervention, the $LSTM_{Seq}$ model with 16 hidden dimensions reaches similar performance as the other approaches. While this model does not outperform traditional approaches regarding prediction accuracy, it provides the potential for further adapting intervention. As the model is able to predict the cluster label at each time step, it is possible to provide the intervention at different points in time for different students depending on how sure the model is about the cluster label of the student. [21] have for example used a simple heuristic to at each time step decide whether the model should continue to see further time steps before outputting a final decision. While exhibiting the same accuracy, classification happened on average at an earlier point in time. This earlier classification allowed to provide targeted interventions sooner.

While the $LSTM_{End}$ model with 16 hidden dimensions is also outperformed by the traditional classifiers at level 7 and 8 of the game, it shows a higher prediction accuracy than the kNN and RF classifiers for the first levels. Our results further demonstrate that the $LSTM_{End}$ model with 16 hidden nodes outperforms the RF classifier regarding the AUC. This is especially important, because the AUC is not biased by imbalanced data sets.

We have also investigated different architectures for the RNN models. Our results demonstrate no large difference in the performance of LSTM and GRU models. However, due to their lower complexity, GRU models are more efficient and take less time to train the LSTM models. While this was not an issue for our small data set, it should be considered when training on larger data sets.

Due to the relatively small number of samples, we were able to only train shallow models with one or two hidden layers, not fully exploiting the advantages of deep learning. Furthermore, we also had to keep the number of dimensions of the hidden state low. However, the results achieved on our small data set are promising and we assume that the RNN models would perform even better on larger data sets.

In the future, we plan to design and test targeted interventions for the different clusters. Furthermore, we will collect a substantially larger data set to enable the training of deep neural networks using raw feature input only. Finally, we also plan to design and train the classifier such that scaffolded interventions can be delivered.

6. REFERENCES

- [1] B. Akram, W. Min, E. N. Wiebe, B. W. Mott, K. Boyer, and J. C. Lester. Improving Stealth Assessment in Game-based Learning with LSTM-based Analytics. In *Proc. EDM*, pages 208–218, 2018.
- [2] S. Amershi and C. Conati. Combining Unsupervised and Supervised Classification to Build User Models for Exploratory Learning Environments. *Journal of Educational Data Mining*, pages 18–71, 2009.
- [3] G. Barata, S. Gama, J. Jorge, and D. Gonçalves. Early Prediction of Student Profiles Based on Performance and Gaming Preferences. *IEEE Transactions on Learning Technologies*, 9(3):272–284, 2016.

- [4] A. F. Botelho, R. S. Baker, and N. T. Heffernan. Improving sensor-free affect detection using deep learning. In *Proc. AIED*, pages 40–51, 2017.
- [5] H. Cen, K. R. Koedinger, and B. Junker. Is Over Practice Necessary? -Improving Learning Efficiency with the Cognitive Tutor through Educational Data Mining. In *Proc. AIED*, pages 511–518, 2007.
- [6] H. Cen, K. R. Koedinger, and B. Junker. Comparing Two IRT Models for Conjunctive Skills. In *Proc. ITS*, pages 796–798, 2008.
- [7] F. Chollet et al. Keras. <https://keras.io>, 2015.
- [8] A. T. Corbett and J. R. Anderson. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *UMUAI*, 4(4):253–278, 1994.
- [9] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell. Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):677–691, 2017.
- [10] Y. Fang, K. Shubeck, A. Lippert, Q. Cheng, G. Shi, S. Geng, J. Gatewood, S. Chen, C. Zhiqiang, P. Pavlik, J. Frijters, D. Greenberg, and A. Graesser. Clustering the Learning Patterns of Adults with Low Literacy Skills Interacting with an Intelligent Tutoring System. In *Proc. EDM*, pages 348–384, 2018.
- [11] L. Fratamico, C. Conati, S. Kardan, and I. Roll. Applying a Framework for Student Modeling in Exploratory Learning Environments: Comparing Data Representation Granularity to Handle Environment Complexity. *International Journal of Artificial Intelligence in Education*, 27(2):320–352, 2017.
- [12] C. Geigle and C. Zhai. Modeling MOOC Student Behavior With Two-Layer Hidden Markov Models. In *Proc. L@S*, pages 205–208, 2017.
- [13] J. P. González-Brenes and J. Mostow. Topical Hidden Markov Models for Skill Discovery in Tutorial Data. *NIPS - Workshop on Personalizing Education With Machine Learning*, 2012.
- [14] T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(1):1–14, 1997.
- [15] S. Kardan and C. Conati. A Framework for capturing distinguishing user interaction behaviours in novel interfaces. In *Proc. EDM*, pages 159–168, 2011).
- [16] T. Käser, A. G. Busetto, B. Solenthaler, J. Kohn, M. von Aster, and M. Gross. Cluster-Based Prediction of Mathematical Learning Patterns. In *Proc. AIED*, pages 389–399, 2013.
- [17] T. Käser, N. R. Hallinen, and D. L. Schwartz. Modeling Exploration Strategies to Predict Student Performance Within a Learning Environment and Beyond. In *Proc. LAK*, pages 31–40, 2017.
- [18] T. Käser, S. Klingler, A. G. Schwing, and M. Gross. Beyond Knowledge Tracing: Modeling Skill Topologies with Bayesian Networks. In *Proc. ITS*, pages 188–198, 2014.
- [19] T. Käser and D. L. Schwartz. Detection and Analysis of Inquiry Strategies related to Transfer and Academic Achievement. *Journal of Artificial Intelligence in Education*, Under Review.
- [20] J. S. Kinnebrew, K. M. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students’ learning behavior patterns. *Journal of Educational Data Mining*, 5(1), 2013.
- [21] S. Klingler, T. Käser, A. G. Busetto, B. Solenthaler, J. Kohn, M. von Aster, and M. Gross. Stealth Assessment in ITS - A Study for Developmental Dyscalculia. In *Proceedings of the Int. l Conference on Intelligent Tutoring Systems (ITS)*, pages 79–89, 2016.
- [22] T. Lange, V. Roth, M. L. Braun, and J. M. Buhmann. Stability-based validation of clustering solutions. *Neural Comput.*, 16(6):1299–1323, 2004.
- [23] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning Word Vectors for Sentiment Analysis. In *Proc. HLT*, pages 142–150, 2011.
- [24] R. E. Mayer. Should there be a three-strikes rule against pure discovery learning? The case for guided methods of instruction. *American Psychologist*, pages 14–19, 2004.
- [25] S. Mojarad, A. Essa, S. Mojarad, and R. S. Baker. Data-Driven Learner Profiling Based on Clustering Student Behaviors: Learning Consistency, Pace and Effort. In *Proc. ITS*, pages 130–139, 2018.
- [26] Z. A. Pardos, C. Hu, P. Meng, M. Neff, and D. Abrahamson. Classifying Learner Behavior from High Frequency Touchscreen Data Using Recurrent Neural Networks. In *Proc. UMAP*, pages 317–322, 2018.
- [27] Z. A. Pardos, S. Tang, D. Davis, and C. V. Le. Enabling Real-Time Adaptivity in MOOCs with a Personalized Next-Step Recommendation Framework. In *Proc. Learning @ Scale*, pages 23–32, 2017.
- [28] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance Factors Analysis - A New Alternative to Knowledge Tracing. In *Proc. AIED*, pages 531–538, 2009.
- [29] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. *Proc. ICML*,, pages 727–734, 2000.
- [30] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Proceedings of NIPS*, pages 505–513, 2015.
- [31] J. L. Sabourin, L. R. Shores, B. W. Mott, and J. C. Lester. Understanding and predicting student self-regulated learning strategies in game-based learning environments. *International Journal of Artificial Intelligence in Education*, 23(1):94–114, 2013.
- [32] R. Sawyer, J. Rowe, R. Azevedo, and J. Lester. Filtered Time Series Analyses of Student Problem-Solving Behaviors in Game-based Learning. In *Proc. EDM*, pages 229–238, 2018.
- [33] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, 2016.
- [34] Y. Wang and J. Beck. Class vs. Student in a Bayesian Network Student Model. In *Proc. AIED*, pages 151–160, 2013.
- [35] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized Bayesian Knowledge Tracing Models. In *Proc. AIED*, pages 171–180, 2013.

One minute is enough: Early Prediction of Student Success and Event-level Difficulty during a Novice Programming Task

Ye Mao

North Carolina State Univ.
Raleigh, NC, USA
ymao4@ncsu.edu

Thomas W. Price

North Carolina State Univ.
Raleigh, NC, USA
twprice@ncsu.edu

Rui Zhi

North Carolina State Univ.
Raleigh, NC, USA
rzhi@ncsu.edu

Tiffany Barnes

North Carolina State Univ.
Raleigh, NC, USA
tmbarnes@ncsu.edu

Farzaneh Khoshnevisan

North Carolina State Univ.
Raleigh, NC, USA
fkhoshn@ncsu.edu

Min Chi

North Carolina State Univ.
Raleigh, NC, USA
mchi@ncsu.edu

ABSTRACT

Early prediction of student difficulty during long-duration learning activities allows a tutoring system to intervene by providing needed support, such as a hint, or by alerting an instructor. To be effective, these predictions must come early and be highly accurate, but such predictions are difficult for open-ended programming problems. In this work, Recent Temporal Patterns (RTPs) are used in conjunction with Support Vector Machine and Logistic Regression to build robust yet interpretable models for early predictions. We performed two tasks: to predict student *success* and *difficulty* during one, open-ended novice programming task of drawing a square-shaped spiral. We compared RTP against several machine learning models ranging from the classic to the more recent deep learning models such as Long Short Term Memory to predict whether students would be able to complete the programming task. Our results show that RTP-based models outperformed all others, and could successfully classify students after just *one* minute of a 20-minute exercise (students can spend more than 1 hour on it). To determine when a system might intervene to prevent incompleteness or eventual dropout, we applied RTP at regular intervals to predict whether a student would make progress within the next five minutes, reflecting that they may be having difficulty. RTP successfully classified these students needing interventions over 85% of the time, with increased accuracy using data-driven program features. These results contribute significantly to the potential to build a fully data-driven tutoring system for novice programming.

1. INTRODUCTION

Modeling student cognitive processes is highly complex since it is influenced by many factors such as motivation, apti-

tude, and learning habits. This is especially challenging for computer-based programming environments, because of the open-ended nature of programming. In this study, we focus on two important types of student modeling tasks to improve student experience in computer-based programming environments: 1) whether students will eventually succeed, and 2) at any given time, whether students need intervention. The interventions are more effective as we can predict earlier. Indeed, student modeling has been studied extensively for well-defined domains like algebra or physics [18, 17, 22]. For an ill-defined domain like programming, there are no pre-defined steps that students must take to complete a given program. Thus, it is hard to map the observations from students step by step. This makes the step-aligned models, like Bayesian Knowledge Tracing (BKT) [10], inappropriate for the programming domain.

Analyzing programming data often requires a way to represent a student's current state on a given problem. In the domain of programming, a student's state is typically represented by their current code, called a code-state. However, this representation leads to very large and poorly connected state spaces [14, 31], which makes it difficult to compare students and apply data-driven methods. Thus, in this study, we transformed student click-like log files into fixed feature sets. As shown in our prior work [38], this feature-state representation dramatically reduces the size of an open-ended programming state space, while creating semantically meaningful states. In this study, we explore both expert feature (EF) and data-driven feature (DDF) sets, and further compare their robustness on the two prediction tasks.

In recent years, deep learning models, specifically Recurrent Neural Networks (RNNs) and RNN-based models such as Long Short Term Memory (LSTM) [13] and gated recurrent unit (GRU) [9], have been shown to achieve state-of-the-art results in many applications with multivariate time series data including educational data. Such models enjoy several nice properties such as strong prediction of performance through deep hierarchical feature construction as well as the ability to effectively capture long-term temporal dependencies in time series data. Despite their extensive applications and great success, the open-ended nature in programming

Ye Mao, Rui Zhi, Farzaneh Khoshnevisan, Thomas Price, Tiffany Barnes and Min Chi "One minute is enough: Early Prediction of Student Success and Event-level Difficulty during Novice Programming Tasks" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 119 - 128

state-space and various time intervals can pose enormous challenges for deep learning. More importantly, these deep learning models are often treated as “black box” models because of the lack of interpretability – they are particularly difficult to understand because of their non-linear nature.

On the other hand, Temporal Pattern-based approaches are designed to extract *interpretable, meaningful* temporal patterns directly from irregularly-sampled multivariate time series data. Recently, significant efforts have been made to develop and apply various pattern-based approaches to healthcare and shown to be effective. [7, 21]. However, as far as we know, temporal Pattern-based approaches have not been extensively investigated in the field of educational data mining especially for programming. In this work, we will directly compare one Temporal Pattern-based approach, Recent Temporal Patterns (RTPs) [7] against a series of baseline models including three classic machine learning models: k-Nearest Neighbors (KNN), support vector machines (SVM), and Logistic Regression (LR), and a state-of-the-art deep learning model: LSTM. More specifically, we compare these approaches on two early prediction tasks. Overall, we show that RTPs outperform all baseline models including LSTM on both early prediction tasks and more importantly, RTPs can generate quite reliable predictions with only the first *one* minute data. Additionally, RTPs can discover interpretable, meaningful temporal patterns that would be informative for domain experts or educators.

Our main contributions are summarized as follows: 1) To the best of our knowledge, this is the first attempt to apply RTP mining to extract student programming temporal patterns and compare it with several baseline models, including deep learning. 2) We run extensive experiments evaluating RTP and various baseline models on both the task of early prediction for student success and that of early prediction for student difficulty, while most prior research mainly focused on one or the other but not both. 3) We explored their robustness and the effectiveness of using EF vs. DDF for the programming system on both early prediction tasks. 4) We identify interpretable, meaningful temporal patterns that can be informative for domain experts.

2. BACKGROUND, CONTEXT, & DATA

2.1 Modeling Student Learning

Student modeling has been extensively explored as a series of approaches have been proposed [10, 34, 8, 24] to better understand and model student learning process. Among them, Bayesian Knowledge Tracing (BKT) [10] is one of the most widely investigated student modeling approaches. It models a student’s performance in solving problems related to a given concept using a binary variable (i.e., correct, incorrect) and continually updates its estimation of the student’s learning state for that concept. BKT and BKT-based models have been shown to be effective in many student modeling tasks, such as predicting students’ overall competence [22], predicting the students’ next-step responses [37, 6, 23, 18], and the prediction of post-test scores [16, 19]. However, in this study, BKT-based models cannot be directly applied to our open-ended programming tasks, because of the adversity of mapping students’ time-various actions step by step.

In recent years, extensive research has been conducted on

deep learning models, especially Recurrent Neural Network (RNN) or RNN-based models such as LSTM in the field of educational data mining [25, 33, 15, 35, 36, 17]. In our prior work, we have shown that LSTM has superior performance on the early prediction of student learning gain compared with classic BKT-based models [19]. For the task of predicting students’ responses to exercises, LSTM was shown to outperform conventional BKT [25] and Performance Factors Analysis [24]. However, RNN and LSTM did not always have better performance when the simple, conventional models incorporated other parameters [15, 35].

While most of the previous studies on student modeling focus on predicting students’ success and failure in the next-step attempt, some research has used student-tutor interaction data to predict student post-test scores [11, 30]. In this work, we explore the early prediction of both student success and difficulty. As far as we know, none of the previous studies have explored both prediction tasks for computer-based programming systems.

2.2 Programming and Help-seeking in iSnap

In this work, we analyze data from *iSnap*, a block-based novice programming environment [26]. iSnap is an extension of Snap! [12], which allows students to easily create interactive, 2D applications, such as apps and games. iSnap provides students with on-demand, next-step programming hints, which are generated automatically from student data with the SourceCheck algorithm [28]. In addition, iSnap collects detailed interaction data as students work, including complete snapshots of all student code, allowing us to perform detailed time series analysis. iSnap’s data-driven hints also offer a useful motivation for this work. Prior analysis of student help-seeking behavior in iSnap suggests that few students ask for help when they need it, especially lower-performing students [29]. This is consistent with help avoidance reported in other tutoring systems [2, 4]. Prior works suggest that a number of factors lead to this help avoidance in iSnap, including lack of trust in the system, a desire for independence, and a lack of awareness of their own need for help [27]. Effective help-seeking is a metacognitive skill that many students need additional training to develop [3]. A system that could detect or predict student difficulty times could offer interventions such as automated help or instructor alerts. However, prior work suggests that such proactive intervention is most effective when systems assess the student’s likelihood to succeed [20], rather than responding to all errors with help messages [1]. In this work, we present a data-driven approach capable of making early and accurate predictions of student success, which can enable a variety of interventions, such as iSnap hints.

2.3 Dataset

Our datasets were collected from students using iSnap in an “introduction to computers” course for non-majors, held at a research university, from the Spring 2016 (S16), Fall 2016 (F16), Spring 2017 (S17), and Fall 2017 (F17) semesters. We excluded students who requested hints since hints may alter students’ problem-solving patterns, and our remaining data contained code traces from 65, 38, 29, and 39 students from S16, F16, S17, and F17 respectively. Each *code trace* consisted of a sequence of timestamped *snapshots* of student code. We chose one “Squirrel” assignment to explore

in-depth, where students must write a procedure to draw a spiral with square corners. Common solutions contain 7-10 lines of code and use procedures, loops, variables, and arithmetic operators. In our previous work, we presented a *feature-state* representation that defines a student's state by the presence or absence of specific features of a correct solution [38]. Each feature describes a distinct property of correct solution code and may specify a necessary code structure or required program output. We defined *feature-state* as the presence or absence of each feature in a student's code (e.g. the feature-state "1101000" indicates Features 1, 2 and 4 are complete, while Features 3,5,6, and 7 are missing). In the same work, we implemented an algorithm to identify data-driven code features directly from student solutions, extracting 11 *data-driven features (DDF)*. We also had experts systematically define 7 *expert features (EF)* for the Spiral assignment. Using the expert feature definitions, we manually tagged each student's snapshot and had 6,339 tagged snapshots from 38 traces in F16. Our work showed that the data-driven features and expert features had moderate agreement on whether a student is in the same feature state or different states for F16. Additionally, we implemented an automated expert feature detector to detect the 7 expert features automatically. We used the detector to tag S16, S17, and F17 student snapshots with expert feature-states. On the F16 dataset, the expert feature detector had an agreement of 0.861 with the manually-tagged expert features. In all, we have 31,064 tagged snapshots from 171 traces from S16, F16, S17, and F17 semesters. Finally, we used a smoothing function to the tagged traces in which periods of rapid feature-state changes, defined as transitioning back and forth between two features within a 5 snapshot window, to set the values of a subsequent period after the variance. This process aims to reduce noise generated by actions in Snap! that do not represent meaningful feature-state changes (e.g. a student drags the code to a different position in the environment).

3. TWO EARLY PREDICTION TASKS

In this work, we explore two different early prediction tasks: the trajectory-level early prediction for student success and the event-level early prediction for student difficulty.

3.1 Trajectory-Level: Student Success

We classify the students who finished the programming assignment in one hour or less and got full credit as *successful*, and those who either failed to complete the assignment or submitted it within one hour as *unsuccessful*. Thus, each *trajectory* is assigned one ground truth label based on whether the student finished the assignment successfully or unsuccessfully. As a result, we refer to this task as *trajectory-level* early prediction task for student success. Based on this definition, 59 of 171 students are in the *successful* group, and the remaining 112 are in the *unsuccessful* group.

To predict student success, we are given the first *up to n* minutes of a student's sequence data and our goal is to predict whether the student will successfully complete the programming assignment at any given point in the remaining of the sequence. To conduct this task, we left-aligned all the students' trajectories by their starting times and our *observation window* (the part of data used to train and test different machine learning models) includes the sequences from

the very beginning to the first *n* minutes. If a student's trajectory is less than *n* minutes, our observation window will include his/her entire sequence *except* the last one.

3.2 Event-Level: Student Difficulty

We define that a student is experiencing some sort of difficulty if at *any given moment*, a student fails to make any progress on his/her incomplete or imperfect answer in *the next five minutes*. Failure to make any progress in five minutes (on a supposed-to-be 20-minute programming task) reflects that the student may be experiencing some difficulty. Identifying the moment that a student is experiencing difficulty would allow us to determine when a system could intervene to address difficulty or prevent eventual dropout.

Because we predict whether a student is experiencing difficulty *moment by moment*, we refer to this early prediction task as *event-level* early prediction. More specifically, for a given moment, *n* minutes after starting time, we classify the students who failed to make any progress in the next five minutes as the *intervention* group, and those who made some progress as the *non-intervention* group. Note that we are not considering students who have already completed the training in *n* minutes and they are not assigned to either group. In short, for the event-level early prediction for student difficulty, our observation window contains the first *up to n* minutes of a student's sequence data and our goal is to predict whether the student will experience any difficulty and need intervention in the *next five minutes*.

4. RTP MINING

Our dataset can be represented as a set of trajectories, $X = \{x_1, x_2, \dots, x_N\}$ where $N = 171$ is the total number of trajectories, one per student. It is composed of multivariate irregular time series data in that a student i 's trajectory x_i consists of a sequence of events: $x_i = \{x_i^1, \dots, x_i^{T_i}\}$, where x_i^t represents the student's code-records at time step t . We have $x_i^t \in \mathbb{R}^D$, where D is the number of predefined attributes/features and each attribute is a binary variable indicating whether a feature is present or not: $\mathbb{R} \in \{0, 1\}$. T_i is the length of the trajectory x_i ; for different students, T_i varies. Each x_i is associated with a trajectory-level label (e.g. student success) or a series of moment by moment event-level labels (e.g. student difficulty), denoted as $Y = \{y_1, y_2, \dots, y_{T_i}\}$, where $y_i \in \{0, 1\}$. It is important to note that in the trajectory-level student success prediction, we treat students who are unsuccessful as the task of interest; thus, they are assigned to be 1 because it is more important for our model to classify and recognize the unsuccessful students as soon as possible. Similarly, for the event-level student difficulty task, we treat students who need intervention as the task of interest, and thus are assigned as 1.

Generally speaking, our RTP mining is conducted by following four steps: 1) Convert binary time-series variables into time interval sequences using Knowledge-based Temporal Abstraction, as described in the next section. 2) Extract frequent recent temporal patterns from different classes of data (i.e. 0 and 1). 3) Transform each x_i into a fixed-size binary feature vector v_i , where the size of vector corresponds to the number of frequent RTPs from Step 2. 4) Build the model on the binary matrix generated in Step 3 to predict the outcomes.

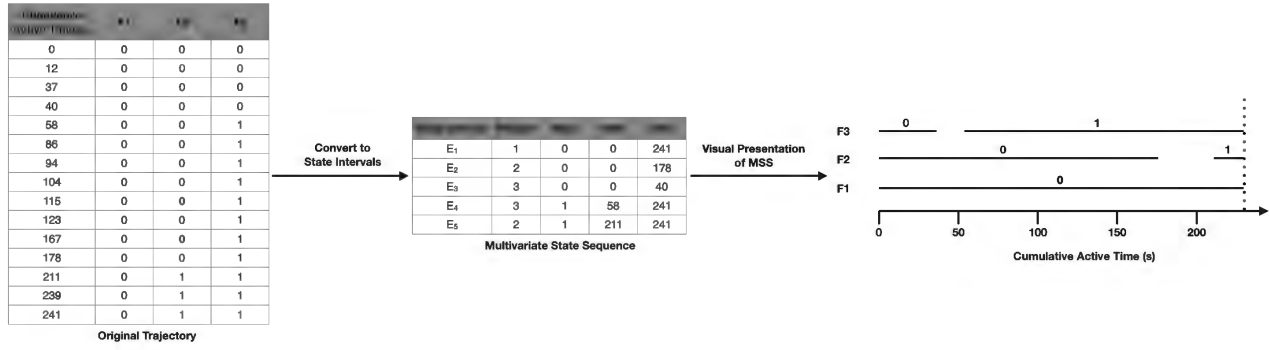


Figure 1: An example of conversion from the original data into a Multivariate State Sequence with three temporal features.

Typically, temporal abstraction involves defining 1) discretized data, 2) multivariate state sequences, 3) temporal relations, and 4) temporal patterns [32]. Our binary data are already discretized, so we describe the remaining three steps.

4.1 Knowledge-based Temporal Abstraction

Multivariate State Sequences: The middle table in Fig. 1 demonstrates how binary data can be converted into Multivariate State Sequences (MSS), z_i where each row presents a state interval, E , extracted from the student's trajectory. We denote a **State** S as (F, V) , where F is a temporal feature and V is the value for feature F at a given time and the **State Interval** E is denoted as (F, V, s, e) , where s and e refer to the *start* and *end* times of the state (F, V) . Thus, we can convert each student's data x_i into a corresponding MSS z_i by sorting all the state intervals by their start times:

$$z_i = \langle E_1, E_2, \dots, E_n \rangle : E_j.s \leq E_{j+1}.s, j \in \{1, \dots, n-1\}$$

Note that we also define $z_i.end$ as the end of the last state interval in the MSS, i.e. $E_n.e$. For example, the right figure of Fig. 1 is a visualization of the MSS z_i , and $z_i.end$ is 241. Applying this method on all $x_i \in X$ transforms X into a set of MSSs: $Z = \{z_1, z_2, \dots, z_N\}$.

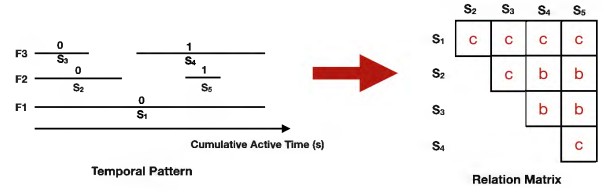


Figure 3: A temporal pattern P with 5 states $\langle S_1 = (F1, 0), S_2 = (F2, 0), S_3 = (F3, 0), S_4 = (F4, 1), S_5 = (F2, 1) \rangle$ and temporal relations presented by half matrix R

Fig. 2. Thus, the two temporal relations, before (b) and co-occurs (c), between two instantaneous events E_i and E_j , are defined as :

- E_i **before** (b) E_j : When E_i ends before the start of E_j ($E_i.e < E_j.s$).
- E_i **co-occurs** (c) with E_j : When E_i and E_j have some overlap ($E_i.s \leq E_j.s \leq E_i.e$).

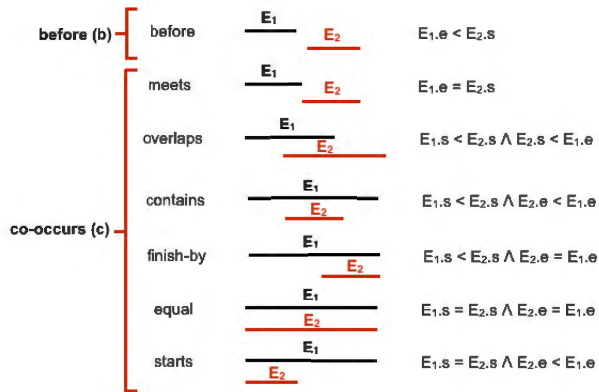


Figure 2: Allen's 7 basic temporal relations, grouped as before, or co-occurs.

Temporal Relations: We define two temporal relations, based on Allen's 7 basic temporal relations between states[5], grouping the last six into one *co-occurs* relation as shown in

Temporal Patterns: Temporal patterns are generated by combining states and temporal relations (before (b) and co-occurs (c)) to describe temporal dependencies in data. More specifically, for n states $\langle S_1, \dots, S_n \rangle$, we define the corresponding temporal pattern: $P = (\langle S_1, \dots, S_n \rangle, R)$, where R is an upper triangular matrix of relations, with $R_{i,j} \in b, c$ corresponding to the relation between S_i and S_j . The size of temporal pattern P is determined by the number of states S it contains. For example, a 5-pattern with three temporal features is shown in Fig. 3, where each state is an abstraction of a variable and the half matrix on the right shows the temporal relations between each pair of states. For example, since $S2 = (\text{Feature 2}, 0)$ happens before $S4 = (\text{Feature 2}, 0)$, $R_{2,4} = b$. In the next step, we describe a method to find the recent temporal patterns (RTPs) from MSSs in Z .

RTP Mining: We selected the RTP mining algorithm proposed by Batal et al. [7], because it incorporates a maximum gap parameter to consider the recency of patterns, it is efficient, and it prevents incoherent patterns. Next, we define

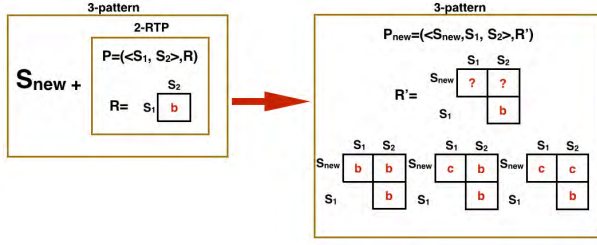


Figure 4: An example of generating 3-patterns out of a single 2-RTP, by appending a new state.

Recent Temporal Patterns (RTP) and briefly explain how they are applied in our work.

Recent Temporal Patterns: First, we call state interval $E = (F, V, s, e)$ a *Recent State Interval* of MSS z_i if: 1) E is the last state interval for feature F ; that is, for all $E' = (F, V', s', e')$, we have $E'.e \leq E.e$; or 2) E is less than g time units away from the end time of the last state interval: $z_i.end$; that is, $z_i.end - E.e \leq g$.

Given an MSS z_i , a temporal pattern $P = (\langle S_1, \dots, S_n \rangle, R)$, and a maximum gap parameter g , we say P is a recent temporal pattern (RTP) in z_i , denoted as $R_g(P, z_i)$, if all 3 of the following conditions hold: 1) z_i contains P , where $P \in z_i$ if: (a) z_i contains all k states of P , and (b) all temporal relations of P are satisfied in z_i ; 2) $S_n = (F_n, V_n)$ matches a recent state interval in z_i ; and 3) Every consecutive pair of states in P maps to a state interval less than g time units apart. That is, each pair of temporal sequences should not be g time units apart. In short, parameter g forces patterns to be close to the end of the sequence z_i , and forces consecutive states to be close to each other.

Mining Algorithm: For each MSS z_i , its outcome $y_i = 1$ when the z_i sequence is unsuccessful/needs intervention, and $y_i = 0$ otherwise. Taking student success classification as an example, we will have two sets of labeled MSSs: $Z_1 = \{z_i : y_i = 1\}$ for all unsuccessful sequences and $Z_0 = \{z_j : y_j = 0\}$ for all successful ones. Given Z_1 , the mining algorithm applies a level-wise search to find frequent RTPs. More specifically, it first starts with all frequent 1-RTPs, and then extends the patterns by adding a new state to each sequence, one at a time, until no new patterns are discovered. That is, at each level k , the algorithm finds frequent $(k+1)$ -RTPs by repeatedly extending k -RTPs through Backward candidate generation, and the Counting phase, as described below.

Backward $(k+1)$ -pattern candidates are generated from a k -pattern $P = (\langle S_1, \dots, S_k \rangle, R)$, by adding a new frequent state, S_{new} , to the beginning of the sequence to create $P' = (\langle S_{new}, S_1, \dots, S_k \rangle, R')$. Then we specify the new before (b) or co-occurs (c) relations R' between S_{new} and all original k states, restricted by the following two criteria: 1) Two state intervals of the same temporal feature cannot co-occur. That is, if $S_{new}.F = S_i.F$ for $i \in \{1, \dots, k\}$, then $R'_{new,i} \neq c$. 2) Since the state sequence in pattern P is sorted by the start time of the states, once a relation becomes *before*: $R'_{new,i} = b$ for any $i \in \{1, \dots, k\}$, all the following relations have to be *before*, so $R'_{new,j} = b$ for $j \in \{i+1, \dots, k\}$.

In the Counting phase, candidate $(k+1)$ -patterns are removed if they do not meet the minimum support threshold by occurring at least σ times as RTP in Z_1 . The same procedure is carried out for Z_0 . Finally, we combine all the frequent RTPs into a final Ω set of RTPs.

Binary Matrix Transformation: We transform each MSS $z_i \in Z$ into a binary vector v_i of size $|\Omega|$, such that each 0 and 1 indicates whether the pattern $P_j \in \Omega$ is a recent temporal pattern in Z_i or not. This will result in a binary matrix of size $N \times |\Omega|$, which represents our original dataset.

Learning Models: Once the binary matrix is built, we apply different machine learning models including KNN, logistic regression (LR), and SVM to perform each target task.

5. EXPERIMENTS

5.1 Seven models in Three Categories

To evaluate the RTP-based models for early prediction of student success and student difficulty, we conducted a series of experiments. For each task, we used grid search to investigate the optimum value for the maximum gap (g) and minimum support (σ) parameters and we have $g = 60$ minutes and $\sigma_0 = 0.2$ for both prediction tasks. For each task, we compared RTP against two categories of baselines: the three *Classic ML models* and *LSTM*. Thus in total, we explored seven classification models in three categories.

Three Classic Machine Learning Models: We explore three classic machine learning models: KNN, LR, and SVM. Since these models do not handle sequence data directly, we used a “Last Value” approach to treat the last measurement of each attribute within the given observation window as the input to train models. Note that “Last Value” approach is also the baseline approach in [7] and many student modeling research. For early prediction settings, we truncated all the sequences in the training dataset in the same fashion as the testing dataset and then applied the Last Value approach on the truncated training dataset. For example, when our observation window is 1 minute, we apply the last value before 1 minute for each sequence and treat them as inputs for each model. For each of the three models, we explored different parameters to obtain the best results, in that, for KNN, we have $k = 10$, for LR we used $L1$ regularization, and for SVM we used *linear* kernel.

One Deep Learning Model: LSTM LSTM is a variation of Recurrent Neural Networks (RNNs) that prevents the vanishing gradient problem among other forms of RNNs [13]. LSTM has a chain-like structure, which enables information to flow among different blocks at different time points. Each block of the LSTM consists of a memory cell state and three gates: Forget, Input, and Output. These three gates interact with each other to control the flow of information. More specifically, the Forget gate determines what information from the previous memory cell state is expired and should be removed; the Input gate selects information from the candidate memory cell state for updating; the Output gate filters the information from the memory cell so that the model only considers information relevant to the prediction task. Therefore, the memory cell plays a crucial role in memorizing previous experiences. In our task, the input is a multivariate temporal sequence from student

work, and the output from the last step is used to make a prediction. We implemented LSTM in Keras with Tensorflow as the back-end engine, and we used one hidden layer with 100 hidden neurons and set the maximum length to accommodate the longest sequence in our data. Typically for LSTM, the whole multivariate time series from student sequence data is used as input data. However, for early prediction, only those events happening within our observation window from each sequence were used. We applied 5-fold cross-validation in order to tune the parameters of the model, including the optimizer, initializer, number of epochs, and number of batches.

Three RTP-based Models: The RTP-based models would first generate the binary matrix through RTP mining and then applied the classical machine learning models, KNN, LR, and SVM, (the same parameter settings as used in classic machine learning models described above) and thus, they are referred as RTP_KNN, RTP_LR, and RTP_SVM, respectively. Prior research on RTPs for prediction have all used entire sequences to extract meaningful temporal patterns. In this work, we explored the effectiveness of RTP for early prediction, by applying the *truncated* training sequences included in observation window to find RTPs. For example, when our observation window is 1 minute, only the first 1 minute of sequences were used for pattern extraction.

5.2 Evaluation Metrics

We evaluated our models using Accuracy, and Recall, F1 Score, and AUC (Area Under ROC curve). Accuracy represents the proportion of students whose labels were correctly identified. Recall tells us what proportion of students who will actually be unsuccessful (or need intervention) were correctly recognized by the model. F1 Score is the harmonic mean of Precision and Recall that sets their trade-off. AUC measures the ability of models to discriminate groups with different labels. Given the nature of the tasks, we mainly use Accuracy and AUC to compare different models. All models were evaluated using 5-fold cross validation.

6. RESULTS

We present our results in three parts. First, we compare the effectiveness of the three categories of models on trajectory-level early prediction of student success. Second, we explore their performance on event-level early prediction of student difficulty. Finally, we discuss the extracted interpretable and meaningful temporal patterns discovered by RTP mining. For both early prediction tasks, we analyze two feature sets: *expert-based features (EF)* and *data-drive features (DDF)*.

6.1 Trajectory-Level: Student Success

[**Observation Window = 1 min**] Table 1 shows the performance of all models using the first-1-minute training sequences to predict students' success. The first row is the baseline model using simple Majority vote; note that we ignored the Recall and F1-measure of the simple Majority baseline. For the three main categories of models, we reported their performance on both EF and DDF. For Classic ML, KNN with DDF generates the highest scores on Recall (0.955) and F1-measure (0.793), SVM with DDF has the best AUC (0.563), and LR with EF contributes the best Accuracy (0.678). Thus, there is no clear winner among the

Table 1: Student success classification performance for the *one-minute* observation window

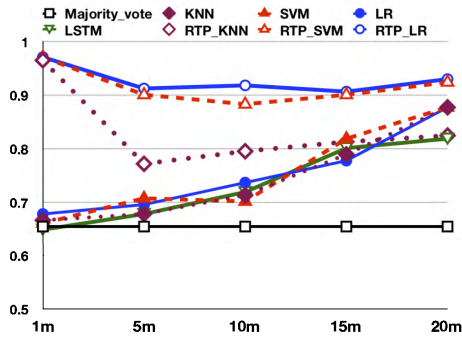
Model	Feature	Accuracy	Recall	F1	AUC
Majority Baseline	EF/DDF	0.655	-	-	0.5
Classic ML	KNN	EF	0.667	0.946	0.788
		DDF	0.673	0.955	0.793
	SVM	EF	0.661	0.946	0.785
		DDF	0.673	0.920	0.786
	LR	EF	0.678	0.938	0.792
		DDF	0.520	0.464	0.559
Deep Learning	LSTM	EF	0.649	0.991**	0.787
		DDF	0.655	0.991**	0.790
RTP	RTP_KNN	EF	0.965	0.973	0.973
		DDF	0.906	0.902	0.927
	RTP_SVM	EF	0.971**	0.955	0.977
		DDF	0.965	0.955	0.973
	RTP_LR	EF	0.971**	0.973	0.978**
		DDF	0.959	0.964	0.969

Note: best model for each group in **bold**, overall best model marked **

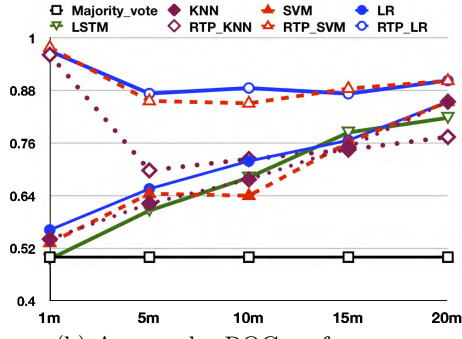
three models here. Between the two types of features, it seems that DDF is slightly better or as good as EF for KNN and SVM, but much worse than EF for LR. Between the two LSTM models, LSTM with DDF works better than LSTM with EF. For RTP-based models, the best Accuracy (0.971) is from RTP_SVM and RTP_LR and both with EF, the best Recall (0.973) is generated by RTP_KNN and RTP_LR and both with EF, the best F1-measure (0.978) comes from RTP_LR with EF again, and the best AUC (0.978) is generated by RTP_SVM with EF. So for RTP-based models, EF generally works better than DDF. However, for RTP_LR and RTP_SVM, the performance of the EF and DDF are very close and competitive.

Finally, across the three categories, RTP-based models have the highest score on every measure except that LSTM has the highest Recall. While all the highest measurements come from using EF, the performance of DDF is very close to EF especially when using LSTM, RTP_LR and RTP_SVM. More importantly, the performance of all the RTP-based models are very high, above 95% on every measure.

[**Observation Window = 1 ~ 20 mins**] Fig. 5 and Fig. 6 report Accuracy and AUC performance for all models using EF and DDF respectively. For each graph, we vary the observation window from the first 1 minute up to the first 20 minutes. Both Fig. 5 and Fig. 6 show that RTP_LR and RTP_SVM were the best models for both using EF and DDF as they stay on the top across all sizes of the observation window. It is not surprising that for the three classic models and LSTM, the longer the observation windows, the better performance they achieve. This is because the training data includes more and more information and closer to their final state. For RTP_KNN, both EF and DDF first decrease dramatically from 1 to 5 minutes and then increase slightly but still, the best performance comes from using only the first *one* minute. For RTP_LR and RTP_SVM, their performances are not only the best but also very steady. The fact that the best prediction comes from using just the first *one* minute of the sequences and using RTP-based models really suggest that *how* students try to solve the problem, what actions they take and in what order in that minute really matters for determining their final success. However, this is only observation from one programming task and more research is needed for further investigation.



(a) Accuracy performance



(b) Area under ROC performance

Figure 5: Student success early prediction on *expert* feature set

When comparing using EF and DDF, Fig. 5 and Fig. 6 shows that the general patterns of performance of different machine learning models are very similar between using EF and DDF with a few exceptions. In general, using EF seems working better than using DDF and the exceptions are: for RTP_LR, and RTP_SVM, the best two models, the performance of using EF and using DDF are very close and sometimes using DDF is even better than using EF.

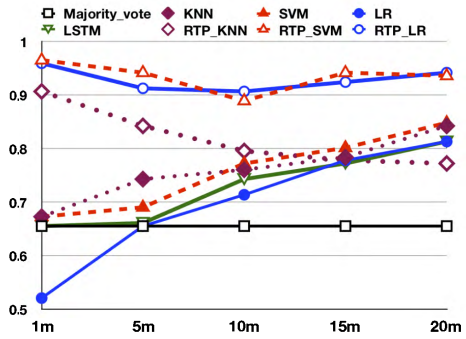
6.2 Event-Level: Student Difficulty

Table 2: Student difficulty classification performance for *one-minute* observation window

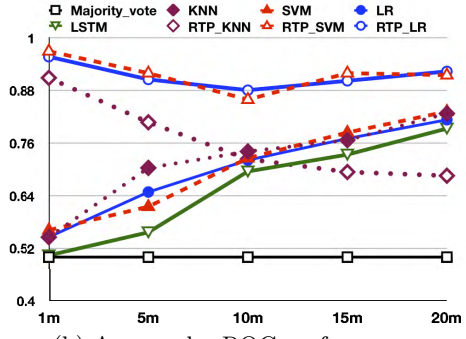
Model	Feature	Accuracy	Recall	F1	AUC
Majority Baseline		0.753	-	-	0.5
Classic ML	KNN EF	0.806	0.238	0.377	0.615
	KNN DDF	0.800	0.214	0.346	0.603
	SVM EF	0.806	0.214	0.353	0.607
	SVM DDF	0.800	0.238	0.370	0.611
	LR EF	0.765	0.238	0.333	0.588
	LR DDF	0.771	0.238	0.339	0.592
Deep Learning	LSTM EF	0.753	0.224	0.344	0.596
	LSTM DDF	0.871	0.269	0.389	0.624
RTP	RTP_KNN EF	0.994**	1**	0.988**	0.996**
	RTP_KNN DDF	0.971	0.976	0.943	0.972
	RTP_SVM EF	0.988	0.976	0.976	0.984
	RTP_SVM DDF	0.971	0.952	0.941	0.964
	RTP_LR EF	0.994**	1**	0.988**	0.996**
	RTP_LR DDF	0.988	1**	0.976	0.992

Note: best model for each group in **bold**, overall best model marked **

[Observation Window = 1 min] Table 2 shows the performance of all models using the first-1-minute-training sequences to predict students' difficulty in the next five min-



(a) Accuracy performance



(b) Area under ROC performance

Figure 6: Student success early prediction on *data-driven* feature set

utes. As with Table 1, it is not very meaningful to present the Recall and F1-measure of the simple Majority baseline (row 1). For Classic ML models, KNN using EF generates the highest scores on every measure but not much better than the other two. In fact, all three models perform very closely regardless of using EF or DDF, and they all performed pretty poorly in that their performances on recall, F1 and AUC are all below 62%. Again for this task, the LSTM models outperform the classic models but not by much. Although LSTM with DDF works better than EF, the resulted models are still not very effective. Finally, for RTP-based models, RTP_KNN and RTP_LR reach the highest scores on every measure. Both of them have the best Accuracy (0.994), the best Recall (1), the best F1-measure (0.988), and the best AUC (0.996). RTP_SVM can make comparable predictions with over 97% on every measure.

In general, the best model among all the seven models uses RTP, which has the highest score on every measure. Comparing the models on different feature sets, we can find that the models with DDF are able to generate very similar results as those with EF. And sometimes, models with DDF had better performance on the first 1 minute, such as LR and LSTM.

[Observation Window = 1 ~ 20 mins] Fig. 7 and Fig. 8 show the results from EF and DDF respectively. We mainly reported the performance of Accuracy and AUC on the observation sets {1m, 5m, 10m, 15m, 20m}. Note that different from trajectory-level early prediction, our majority baseline for event-level early prediction on student difficulty is changing moment by moment and for an observation window n ,

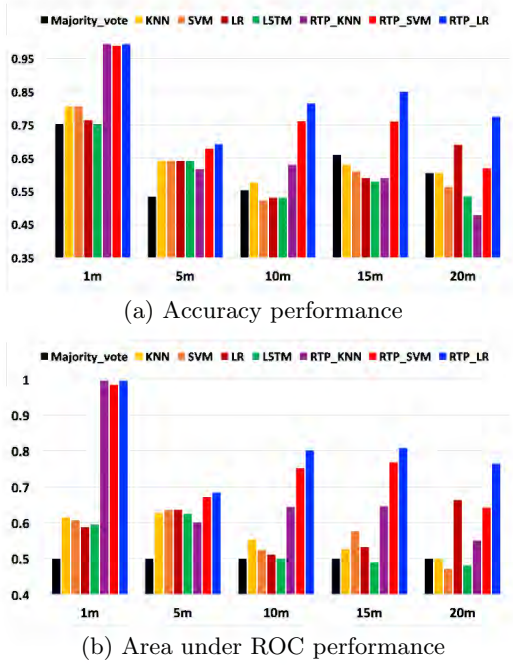


Figure 7: Student difficulty early prediction on *expert* feature set

all the students whose trajectories are shorter than n are excluded from the evaluation. So we have different baseline majority accuracy for different observation window (shown in black bar). For AUC, majority vote would give us a baseline of 0.5. Both Fig. 7 and Fig. 8 show that RTP_LR is the best one among all the models no matter which feature sets were used, with the highest scores on both Accuracy and AUC for all observation windows. Additionally, when comparing RTP_LR with EF and RTP_LR with DDF, we can see that RTP_LR with DDF generally perform much better than RTP_LR with EF. By comparing the results from Fig. 7 and Fig. 8, RTP_LR and RTP_SVM really benefit from using DDF instead of EF in that while RTP_LR and RTP_SVM with DDF perform very closely with RTP_LR and RTP_SVM with EF when the observation window = 1m, the RTP_LR and RTP_SVM with DDF performed much better and more stable than RTP_LR and RTP_SVM with EF on all the following observation windows and the difference are large. For the rest of machine learning models, the difference between using DDF and using EF is not noticeable. Finally, note that Fig. 7 shows when the observation window = 5m, the best performance using EF is 65% accuracy and < 70% of AUC achieved through RTP_LR ($x - axis = 5m$); however, when we use the same observation window, RTP_LR with DDF can achieve the accuracy close to 95% and AUC above 90% (shown in Fig. 8). So overall, for the task of event level early prediction for student difficulty, RTP_LR with DDF consistently achieve the best performance and its performance is pretty steady across different observation windows.

6.3 Knowledge Discovery

One substantial advantage of pattern-based classification over deep learning models is the *interpretability* of the discovered

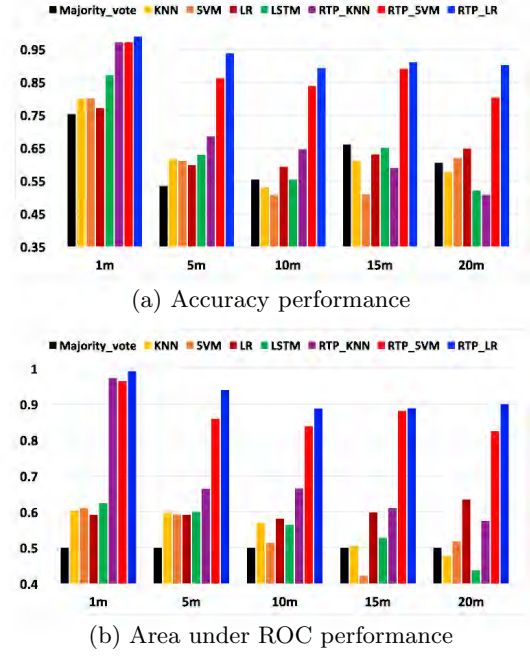


Figure 8: Student difficulty early prediction for *data-driven* feature set

patterns. In RTP, the patterns need to be representative of the original time series data while predictive of the future outcomes. In this study, most of the patterns extracted using RTP mining are in accordance with the student classification tasks and some of them reveal latent patterns towards the progression of student success or difficulty. Table 3 and 4 present a number of interesting patterns and their corresponding support among the training group from the first 20 minutes, where the support of pattern P is calculated as the proportion of students in the dataset which contains P .

Table 3 shows some patterns related to student success. $P_1 - P_4$ describe the frequent patterns found among unsuccessful students, and most of them are related to the feature *MoveVariably*. P_1 , P_2 and P_3 illustrate that students did not complete feature *CreateUseParameterCorrectly*, *RepeatCorrectNumberOfTimes*, or *MoveSquirally* when feature *MoveVariably* has been completed. In P_1 , students may start to work on *DrawAnything* and *MoveSquarelikeThing*, which is observable among 29.5% of unsuccessful students from their sequences in the first 20 minutes. Different from P_1 , students with pattern P_2 and P_3 were probably working on *CustomBlock* before they finished feature *MoveSquirally*. P_4 indicates that students had finished features *MoveVariably* and *MoveSquarelikeThing* but failed at the end, which might be the case that they had done some movements but did not have pen down. And still, neither *CreateUseParameterCorrectly* nor *RepeatCorrectNumberOfTimes* was finished. $P_5 - P_8$ describes the frequent patterns found among successful students, and most of them are related to the feature *RepeatCorrectNumberOfTimes*. For example, P_5 shows that students had finished feature *CustomBlock*, *CreateUseParameterCorrectly* and *RepeatCorrectNumberOfTimes* at some point, and before that, within 20 minutes, they should be working

Table 3: Recent temporal patterns for student success, with observation window = 20m

RTP	If	Then	Support
P_1	(CreateUseParameterCorrectly,0) c (((DrawAnything,0) c (MoveSquarelikeThing,0)) b (MoveVariably,1))	Unsuccessful	0.295
P_2	((CustomBlock,0) b (MoveVariably,1)) c (CreateUseParameterCorrectly,0) c (RepeatCorrectNumberOfTimes,0)	Unsuccessful	0.286
P_3	((CustomBlock,0) b (CreateUseParameterCorrectly,0)) c (RepeatCorrectNumberOfTimes,0) c (MoveSquirally,0) c (MoveVariably,1)	Unsuccessful	0.233
P_4	((DrawAnything,0) b ((MoveSquarelikeThing,1) c (MoveVariably,1))) c (CreateUseParameterCorrectly,0) c (RepeatCorrectNumberOfTimes,0)	Unsuccessful	0.205
P_5	(CustomBlock,0) b (CustomBlock,1) c (CreateUseParameterCorrectly,1) c (RepeatCorrectNumberOfTimes,1)	Successful	0.729
P_6	(CustomBlock,0) c (MoveSquarelikeThing,0) b (MoveSquarelikeThing,1) c (RepeatCorrectNumberOfTimes,1)	Successful	0.542
P_7	(DrawAnything,0) b (CreateUseParameterCorrectly,0) b (CreateUseParameterCorrectly,1) c (MoveVariably,1)	Successful	0.423

Table 4: Recent temporal patterns for student difficulty, with observation window = 20m

RTP	If	Then	Support
P_1	((((DrawAnything,0) c (MoveSquarelikeThing,0)) b (CustomBlock,1)) c (RepeatCorrectNumberOfTimes,0) c (MoveSquirally,0))	Intervention	0.372
P_2	((((CustomBlock,0) c (DrawAnything,0)) b ((CustomBlock,1) c (MoveVariably,1))) c (MoveSquirally,0))	Intervention	0.302
P_3	((DrawAnything,0) c (MoveSquarelikeThing,0)) b ((CustomBlock,1) c (DrawAnything,1)) c (MoveSquirally,0)	Intervention	0.279
P_4	((MoveSquarelikeThing,0) b ((MoveSquarelikeThing,1) c (MoveSquirally,1))) c (RepeatCorrectNumberOfTimes,0)	Non-intervention	0.461
P_5	((((DrawAnything,0) c (MoveSquarelikeThing,0)) b (MoveSquirally,1)) c (RepeatCorrectNumberOfTimes,0))	Non-intervention	0.422
P_6	(DrawAnything,0) c (MoveSquarelikeThing,0) c (CreateUseParameterCorrectly,0) c (RepeatCorrectNumberOfTimes,0) c (CustomBlock,1)	Non-intervention	0.320

on *CustomBlock*. Actually, for the students in successfully group, they should be able to finish all the features in one hour. Glancing over the extracted patterns among the successful group, we can see that most of the students ended up completing at least two important features, like patterns P_6 and P_7 in the Table 3, in the first 20 minutes. It is important to note that these patterns are only discovered among the successful group.

Table 4 shows some patterns related to student difficulty. $P_1 - P_3$ describe the frequent patterns found among the intervention group, and most of them are related to the feature *CustomBlock*. P_1 describes a pattern that discovered among 37.2% of students who need intervention in the next five minutes. In this case, students had not completed the *DrawAnything* or *MoveSquarelikeThing* features at first, but, within 20 minutes, they completed the *CustomBlock* feature. And at the same time, they did not complete feature *RepeatCorrectNumberOfTimes* or *MoveSquirally*. P_2 and P_3 are quite similar, they indicate that students finished *CustomBlock* along with *DrawAnything* or *MoveVariably* when the feature *MoveSquirally* had not been finished. P_4 to P_6 describe the frequent patterns found among students who do not need intervention, and most of them are related to the feature *MoveSquirally*. $P_4 - P_5$ show that students successfully completed feature *MoveSquirally* and before that, they may work on *MoveSquarelikeThing*. As in P_1 , they have ‘0’ on the feature *RepeatCorrectNumberOfTimes*. Comparing P_1 with P_5 , we can find that instead of having incomplete *DrawAnything* and *MoveSquarelikeThing* before completing *CustomBlock*, students who have incomplete *DrawAnything*, incomplete *MoveSquarelikeThing*, and complete *CustomBlock* at the same time will not need intervention in the next five minutes. Again, these patterns are uniquely inferred among the non-intervention group.

7. CONCLUSIONS

Early prediction of trajectory-level student success and the event-level difficulty during long-term activities are challenging tasks due to the open-ended nature of programming tasks. In this study, we explored the EF identified by domain experts, as well as DDF identified automatically, to build a model that is able to predict student success/difficulty with

high accuracy, and to provide valuable insights for educators. We employed an RTP-based classification framework and compared it with various baselines including classic and deep learning models, in two different tasks including trajectory-level early diagnosis and event-level early prediction. Our results suggest that the RTP-based models consistently outperform the non-temporal classic baselines as well as LSTM in both tasks, at all observation windows from first 1 minute to 20 minutes. Moreover, with only the first-1-minute sequences, RTP-based models can make strong predictions on both tasks. Additionally, by applying the data-driven features, RTP-based models are able to achieve comparable performance on the task of predicting student success. For the task of predicting event-level student difficulty, data-driven features can even improve their predictions further.

As future work, we plan to apply progressive features with multiple values (eg. not started, in progress, complete) to discover more definitive patterns and obtain more advantageous knowledge discovery as well as improved prediction performance. And we are planning to employ different feature transformations other than binary, such as vertical support (i.e. the number of times a pattern occurred in a sequence) or recency measure (i.e. how distant a pattern occurred from the prediction time). Also, this work will be applied to larger groups of students and longer programming tasks, along with integration of more informative features such as intervention and demographic features to develop more robust models. Additionally, we plan to expand our evaluations to longer programs with more complex constructs from both text-based and block-based programming languages.

8. ACKNOWLEDGEMENTS

This research was supported by the NSF Grants 1432156, 1623470, 1651909, and 1726550.

9. REFERENCES

- [1] Hints: Is it better to give or wait to be asked? In *ITS*, volume 6094 LNCS, pages 349–358, 2010.
- [2] V. Aleven and K. R. Koedinger. Limitations of Student Control: Do Students Know When They Need Help? In *ITS*, pages 292–303, 2000.

- [3] V. Aleven, B. M. McLaren, I. Roll, and K. R. Koedinger. Toward Meta-cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. *IJAIED*, 16:101–128, 2006.
- [4] V. Aleven, E. Stahl, S. Schworm, F. Fischer, and R. Wallace. Help seeking and help design in interactive learning environments. *Review of educational research*, 73(3):277–320, 2003.
- [5] J. F. Allen. Towards a general theory of action and time. *Artificial intelligence*, 23(2):123–154, 1984.
- [6] R. S. Baker, A. T. Corbett, and V. Aleven. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *ITS*, pages 406–415, 2008.
- [7] I. Batal, D. Fradkin, J. Harrison, F. Moerchen, and M. Hauskrecht. Mining recent temporal patterns for event detection in multivariate time series data. In *SIGKDD*, pages 280–288. ACM, 2012.
- [8] M. Chi, K. R. Koedinger, G. J. Gordon, P. Jordon, and K. VanLahn. Instructional factors analysis: A cognitive model for multiple instructional interventions. In *EDM*, pages 61–70, 2011.
- [9] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [10] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *UMUAI*, 4(4):253–278, 1994.
- [11] M. Feng, J. Beck, N. Heffernan, and K. Koedinger. Can an intelligent tutoring system predict math proficiency as well as a standardized test? In *EDM*, pages 107–116, 2008.
- [12] D. Garcia, B. Harvey, and T. Barnes. The Beauty and Joy of Computing. *ACM Inroads*, 6(4):71–79, 2015.
- [13] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [14] B. P. Iii, A. Hicks, and T. Barnes. Generating hints for programming problems using intermediate output. In *EDM*. Citeseer, 2014.
- [15] M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing? *arXiv preprint arXiv:1604.02416*, 2016.
- [16] C. Lin and M. Chi. Intervention-bkt: incorporating instructional interventions into bayesian knowledge tracing. In *ITS*, pages 208–218. Springer, 2016.
- [17] C. Lin and M. Chi. A comparisons of bkt, rnn and lstm for learning gain prediction. In *AIED*, pages 536–539. Springer, 2017.
- [18] C. Lin, S. Shen, and M. Chi. Incorporating student response time and tutor instructional interventions into student modeling. In *UMAP*, pages 157–161. ACM, 2016.
- [19] Y. Mao, C. Lin, and M. Chi. Deep learning vs. bayesian knowledge tracing: Student models for interventions. *JEDM*, 10(2):28–54, 2018.
- [20] V. K. Murray R.C. A comparison of decision-theoretic, fixed-policy and random tutorial action selection. *LNCS*, 4053 LNCS:114–123, 2006.
- [21] K. Orphanou, A. Dagliati, L. Sacchi, A. Stassopoulou, E. Keravnou, and R. Bellazzi. Combining naive bayes classifiers with temporal association rules for coronary heart disease diagnosis. In *ICHI*, pages 81–92, 2016.
- [22] Z. A. Pardos and N. T. Heffernan. Modeling individualization in a bayesian networks implementation of knowledge tracing. In *UMAP*, pages 255–266. Springer, 2010.
- [23] Z. A. Pardos and N. T. Heffernan. Kt-idem: Introducing item difficulty to the knowledge tracing model. In *UMAP*, pages 243–254. Springer, 2011.
- [24] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis –a new alternative to knowledge tracing. In *AIED*, pages 531–538, 2009.
- [25] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *NIPS*, pages 505–513, 2015.
- [26] T. W. Price, Y. Dong, and D. Lipovac. iSnap: towards intelligent tutoring in novice programming environments. In *SIGCSE*, pages 483–488, 2017.
- [27] T. W. Price, Z. Liu, V. Catete, and T. Barnes. Factors Influencing Students’ Help-Seeking Behavior while Programming with Human and Computer Tutors. In *ICER*, 2017.
- [28] T. W. Price, R. Zhi, and T. Barnes. Evaluation of a Data-driven Feedback Algorithm for Open-ended Programming. In *EDM*, 2017.
- [29] T. W. Price, R. Zhi, and T. Barnes. Hint Generation Under Uncertainty: The Effect of Hint Quality on Help-Seeking Behavior. In *AIED*, 2017.
- [30] S. Ritter, A. Joshi, S. Fancsali, and T. Nixon. Predicting standardized test scores from cognitive tutor interactions. In *EDM*, pages 169–176, 2013.
- [31] K. Rivers and K. R. Koedinger. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *IJAIED*, 27(1):37–64, 2017.
- [32] Y. Shahar. A framework for knowledge-based temporal abstraction. *Artificial intelligence*, 90(1-2):79–133, 1997.
- [33] S. Tang, J. C. Peterson, and Z. A. Pardos. Deep neural networks and how they apply to sequential education data. In *L@S*, pages 321–324. ACM, 2016.
- [34] K. Tatsuoaka. Rule space: An approach for dealing with misconceptions based on item response theory. *JEM*, 20(4):345–354, 1983.
- [35] K. H. Wilson, Y. Karklin, B. Han, and C. Ekanadham. Back to the basics: Bayesian extensions of irt outperform neural networks for proficiency estimation. *arXiv preprint arXiv:1604.02336*, 2016.
- [36] X. Xiong, S. Zhao, E. Van Inwegen, and J. Beck. Going deeper with deep knowledge tracing. In *EDM*, pages 545–550, 2016.
- [37] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *AIED*, pages 171–180. Springer, 2013.
- [38] R. Zhi, T. W. Price, N. Lytle, Y. Dong, and T. Barnes. Reducing the state space of programming problems through data-driven feature detection. In *EDM (Workshops)*, 2018.

Kappa Learning: A New Item-Similarity Method for Clustering Educational Items from Response Data

Tanya Nazaretsky
Weizmann Institute of Science
Rehovot, Israel
tanya.nazaretsky@weizmann.ac.il

Sara Hershkovitz
The Center for Educational
Technology
Tel Aviv-Yafo, Israel
sarah@cet.ac.il

Giora Alexandron
Weizmann Institute of Science
Rehovot, Israel
giora.alexandron@weizmann.ac.il

ABSTRACT

Sequencing items in adaptive learning systems typically relies on a large pool of interactive question items that are analyzed into a hierarchy of skills, also known as Knowledge Components (KCs). Educational data mining techniques can be used to analyze students response data in order to optimize the mapping of items to KCs, with similarity-based clustering as one of the two main approaches for this type of analysis. However, current similarity-based methods make the implicit assumption that students' performance on items that belong to the same KC should be similar. This assumption holds if the latent trait (mastery of the underlying skill) is relatively fixed during students' activity, as in the context of *testing*, which is the primary context in which these methods were developed and applied. However, in adaptive learning systems that aim for *learning*, and address subject matters such as K-6 Math that consist of multiple sub-skills, this assumption *does not* hold. In this paper we propose a new item-similarity measure, termed *Kappa Learning* (KL), which aims to address this gap. KL identifies similarity between items under the assumption of *learning*, namely, that learners' mastery of the underlying skills changes as they progress through the items. We evaluate KL on data from a K-6 Math Intelligent Tutoring System, with experts' tagging as ground truth, and on simulated data. Our results show that clustering that is based on KL outperforms clustering that is based on commonly used similarity measures (Cohen's Kappa, Yule, and Pearson), and that KL is also superior in the task of discovering the number of KCs.

Keywords

Intelligent Tutoring Systems, Adaptive Learning, Clustering Educational Items, Similarity Measurement

1. INTRODUCTION

Mastery learning [4] is based on the assumption that the domain knowledge can be analyzed into a hierarchy of component skills, with prerequisites between them [9, 10]. This

structure can be used to sequence learning in an Intelligent Tutoring System (ITS) so that students master prerequisite skills before moving to skills that depend upon them [10]. *Cognitive model* is a formal representation of this structure that is encoded into the ITS. It is typically generated in a process that relies on domain experts, learning scientists, and programmers [6].

A significant part of this process is the mapping of question items into the skills that underlie them (skills are also referred to as Knowledge Components, abbreviated KCs¹; in this paper we use the two terms interchangeably). Q-matrix is a standard representation used in Psychometrics to specify the relationships between individual test items and target skills [28]. Generating item-to-skill mapping requires a significant human-labor and expertise [13]. In addition, evidence shows that experts' mapping of items into skills can be significantly inconsistent with students' learning process [20]. Thus, methods that identify the skills underlying each item, or assist human experts in doing so, can optimize the process by increasing its accuracy and reducing human labor [14, 19].

Constructing Q-Matrix from response data is an active research topic. Barnes [3] "mined" students' data to create concept models that can be used to direct learning paths. Examples within the Psychometrics literature include [11, 21, 28]. A Matrix Factorization-based method for Q-matrix construction was proposed in [12], and was later used for enhancing expert-based Q-Matrices [14]. Learning Factor Analysis (LFA) [6] is a combinatorial search algorithm for optimizing the cognitive model while controlling for model complexity. In [22] it was demonstrated that using LFA to refine the human-generated cognitive model of an ITS improves learning gains. Performance Factor Analysis (PFA) [24] reconfigured LFA to enable predictions for individual students with individual skills (LFA assumes all students accumulate learning at the same rate), and also addresses the multiple KCs problem (standard Knowledge Tracing [10] assumes that each item requires one KC; examples of extensions that address multiple skills include [15, 32]). A different approach for 'human-in-the-loop' Student Model Discovery (finding the item-to-skill assignment that best describes students' behavior) was proposed in [27].

In general, there are two approaches for mapping items into

¹in the Psychometrics literature, skills are also referred to as *latent factors* or *constructs*

Tanya Nazaretsky, Sara Hershkovitz and Giora Alexandron "Kappa Learning: A New Item-Similarity Method for Clustering Educational Items from Response Data" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 129 - 138

skills: *Model-based*, and *Similarity-based* [26]. Model-based methods reduce the dimensionality of the problem and try to infer the latent factors (=skills or KCs) that underlie the items. The methods mentioned above fall into this category. Similarity-based methods are based on the assumption that students will tend to have similar performance on items that require the same skill, thus seek to identify the similarity between pairs of items. Examples of methods that are based on item-similarity measures include [2, 26]. The method that we propose in this paper falls into this category as well.

The first phase of item similarity-based methods consists of computing a similarity measure for each pair of items. This measure can be then used to cluster items, which is naturally interpreted as associating the items of a cluster with a single KC. In [26], different measures of item similarity (Pearson, Cohen’s Kappa, Yule, Jaccard, and Sokal) were evaluated on real and simulated data. A different method for identifying the similarity between pairs of items, which is based on Fisher’s Exact Test of independence, was proposed in [2] and was applied to data from an Introductory Physics MOOC. In addition to correct/incorrect information, ‘item-similarity’ can be based also on other behavioral characteristics, such as response-times [5, 26].

The item-similarity methods used in educational data mining for clustering items make the implicit assumption that the latent trait (mastery of the specific skill) is *fixed* during the learning activity that generated the responses (so students’ responses to items that belong to the same KC should be highly correlated). This assumption may be reasonable in the context of *testing* (summative assessment), which is expected to occur *after* the learning process. (In [26], the authors explicitly refer to this shortcoming of the item-similarity measures and mention that by using these methods “we mostly ignore the issue of learning”, p. 17.) However, this assumption *does not* hold in the context of *learning*. In such cases, the correlation between the items might not be a good indication of their similarity (e.g., students will tend to fail on the first items of each KC, and succeed on later ones).

The goal of this research is to address this gap by providing a measure that can capture similarity in the context of *learning*. For that, we propose a new item-similarity measure termed *Kappa Learning* (KL). The main assumption behind KL is that students’ performance on items belonging to the same KC can be increasing, but not decreasing. As we use dichotomous scoring (correct/incorrect on first attempt), we expect that the performance of student s on KC k would take the form of a ‘step’ function, which moves from 0 to 1 when s masters k (*guess* or *slip* may occur, and introduce noise). To quantify that, KL extends the notion of ‘agreement’ in *Cohen’s Kappa* [8].

We first make the assumption that the items are administered to the students in the same order (defined by the instructional designers), but we later explain how our formula naturally generalizes to random or adaptive ordering. We note that we do not assume that all items belonging to the same KC will be presented to students one after the other, or that all the students attempt all the items. On the contrary, we assume that students can skip items, and that

items from different KCs may interleave (as in the data that we analyze), which makes the clustering non-trivial. We then compare a clustering that is based on KL to clustering that is based on the similarity measures evaluated in [26], and show that KL significantly outperforms them (Jaccard and Sokal, which achieved the lowest results in [26], and also on our data, are omitted from the analysis).

The rest of this paper is organized as follows. In Section 2, we present Cohen’s Kappa and our new measure, Kappa Learning. Section 3 describes the clustering method. The details of the empirical setting and the data are provided in Section 4. In Sections 5 and 6 we evaluate the performance of Kappa Learning against standard similarity measures on real and simulated data, respectively. Finally, in Section 7 we discuss the results and suggest directions for future research.

2. COHEN’S KAPPA AND KAPPA LEARNING

2.1 Cohen’s Kappa

Cohen’s Kappa (sometimes abbreviated as Kappa and denoted S_k) is a measure of inter-rater agreement for nominal scales [8].

$$S_k = \frac{P_o - P_e}{1 - P_e} \quad (1)$$

where:

P_o is an *observed* level of agreement

P_e is an *expected* level of agreement

The observed level of agreement is the proportion of the cases the raters agree upon. The expected level of agreement is the proportion of agreement that is expected by chance.

We consider items as raters, learners as subjects to classify, and learners’ responses as classification results. We interpret learner’s correct/incorrect answer to an item (encoded as 1/0) as the rater’s (=item) attempt to identify if the learner has mastered the KC underlying the item. Let us consider a contingency table summarizing learners’ responses to two different items: Q_1 and Q_2 . Assume n learners answered both items. The number of learners in each cell is defined as follows (Table 1):

- a - number of learners answered both Q_1 and Q_2 correctly
- b - number of learners answered Q_1 incorrectly and Q_2 correctly
- c - number of learners answered Q_1 correctly and Q_2 incorrectly
- d - number of learners answered both Q_1 and Q_2 incorrectly
- n - total number of learners ($n = a + b + c + d$)

The number of cases the raters agree upon (the learner gave the same answer to both items) is equivalent to $a + d$. Intuitively, if two different items belong to the same skill, and

Table 1: A contingency table for Q_1 and Q_2 .

	Q_1 correct	Q_1 incorrect	
Q_2 correct	a	b	a + b
Q_2 incorrect	c	d	c + d
	a + c	b + d	n

learner’s mastery of that skill is *fixed* during the learning activity, the learner is expected to answer both items either correctly or incorrectly, depending on whether the skill is mastered or not. So, it follows that:

$$P_o = \frac{a + d}{n}$$

The items are independent in the sense that each item independently ‘rates’ if a learner belongs to a category of learners knowing a particular KC. So we could compute the level of agreement that is expected by chance as a sum of products of marginal probabilities (Table 1).

$$P_e = \frac{(a + b)(a + c) + (b + d)(c + d)}{n^2}$$

By doing substitution of P_o and P_e into Equation 1 and some straightforward simplification we get:

$$S_k = \frac{2(ad - bc)}{(a + b)(b + d) + (a + c)(c + d)}$$

2.2 Kappa Learning: Adjusting Kappa to Accommodate Learning

To accommodate for learning, we give a different interpretation to the notion of ‘agreement’ in Cohen’s Kappa formula, taking into account possible improvement of learner’s skill, or in other words, learning.

We make the following assumptions on the process:

1. The items are presented to the learners in a fixed order².
2. The items belong to k KCs ($k > 1$); Each item belongs to one KC; Items belonging to different KCs may interleave (which makes the clustering non-trivial)
3. Learner’s success on items belonging to the same KC behaves like a ‘step’ function: Before mastering the skill of KC k , the student fails on items of k ; once mastering the skill underlying k , the student succeeds on future items of k (*guess* and *slip* may occur; we assume no ‘forgetting’).

For a pair of items Q_1 , Q_2 , where Q_1 is presented to the learners *before* Q_2 , we define the values in the contingency table (Table 1) as follows:

- a - number of learners who got both items correct, namely mastered the required skills before getting to Q_1 . This is a case of agreement.

²We later explain how this assumption can be removed

- b - number of learners who got the first item incorrect and the second item correct, namely, mastered the required skill after getting to Q_1 , but before getting to Q_2 . **This is a case of agreement, and is where our measure differs from Cohen’s Kappa**
- c - number of learners who got the first item correct and the second item incorrect. This is the only case interpreted as *disagreement*.
- d - number of learners answered both Q_1 and Q_2 incorrectly. This is a case of agreement.
- n - total number of learners ($n = a + b + c + d$)

Based on these, we define P_o and P_e as follows:

$$P_o = \frac{a + b + d}{n}$$

$$P_e = \frac{(a + b)(b + d) + (a + b)(a + c) + (b + d)(c + d)}{n^2}$$

By doing substitution of P_o and P_e into Equation 1 and some straightforward simplification we get:

$$S_{kl} = \frac{(ad - bc)}{(a + c)(c + d)} \quad (2)$$

We call this measure Kappa Learning and denote it S_{kl} . The values of both Kappa and Kappa Learning range between -1 and $+1$, where 0 means independence, and $+1$ means perfect agreement. In Kappa it is achieved when both c and b are equal to 0 . In Kappa Learning, perfect agreement is achieved when c equals 0 .

3. METHOD

3.1 Similarity Measures

To evaluate the performance of Kappa Learning (denoted S_{kl}), we compare it to the following similarity measures:

- S_k : Cohen’s Kappa inter-rater agreement
- S_p : Pearson product-moment correlation coefficient
- S_y : Yule coefficient of association

Cohen’s Kappa (see also in Subsection 2.1) coefficient is defined as:

$$S_k = \frac{2(ad - bc)}{(a + b)(b + d) + (a + c)(c + d)} \quad (3)$$

Pearson product-moment coefficient is a measure of linear correlation between two variables. When applied to dichotomous data, the Pearson correlation coefficient returns the phi (ϕ) coefficient. So, in terms of a , b , c and d (Table 1) the value of S_p is computed as follows:

$$S_p = \frac{(ad - bc)}{\sqrt{(a + c)(a + b)(b + d)(c + d)}} \quad (4)$$

Yule coefficient of association is a measure of colligation between two binary variables and it is commonly used for

analyzing scores in Item Response Theory (IRT). It is the number of pairs in agreement (ad) minus the number in disagreement (cb) over the total number of paired observations and it is defined as:

$$S_y = \frac{(ad - bc)}{(ad + bc)} \quad (5)$$

All three measures range from minus unity to unity, where 1 indicates perfect agreement, -1 indicates perfect disagreement and 0 indicates no relationship [30]. A thorough evaluation of these measures as means for clustering items in an interactive learning environment was done by Řihák and Pelánek [26] (they analyzed the most appropriate measures among the 76 measures analyzed in [7]).

We follow a similar methodology to the one proposed in [26], described below, and demonstrate that Kappa Learning outperforms the other measures.

3.2 Process

Our process has two main steps: 1) Cluster the items based on the four similarity measures (Kappa Learning, and the three reference measures). 2) Compare the goodness-of-fit of the clusterings computed in step 1.

Step 1. Computing the clustering includes the following sub-steps (per similarity measure):

1. From students' performance data, we compute *user-based* item similarity matrix, denoted $M1$. $M1[i, j]$ contains the result of the relevant similarity measure for items q_i and q_j .
2. Compute *item-based* distance matrix from the user-based similarity matrix $M1$. The rationale is that for a pair (q_i, q_j) , if q_i and q_j are similar (i.e., belong to the same KC), they should have a similar distance to a third item q_k (whether it is in the same KC or not). This incorporates more information into the similarity between the items, which should improve the accuracy of the clustering [26]. We denote the *item-based* distance matrix with $M2$. Two standard metrics are used for computing $M2$: Pearson and Euclidean.
3. Two clustering algorithms are applied on $M2$: K-Means and Ward's Hierarchical [17]. The number of clusters is derived from the hierarchal Knowledge Tree defined by content experts (see Subsection 4.2).

Step 2. Per clustering, we use Adjusted Rand Index (ARI) [16, 25] to measure the goodness-of-fit against ground truth – experts' mapping of the items into Knowledge Components.

ARI is a common measure for comparing the similarity between two clusterings. In ARI, a similarity is interpreted as the number of pairs of items on which the clusterings 'agree', adjusted for the amount of agreement 'by chance'.

To be concrete, assume C is a dataset which contains m items, with two clusterings of C into k clusters, denoted C_1 and C_2 . For a pair of items (i_1, i_2) , C_1 and C_2 'agree' on

(i_1, i_2) iff i_1 and i_2 are either assigned to the same cluster, or to different clusters, in both C_1 and C_2 .

To evaluate the level of agreement between C_1 and C_2 , we define a contingency table with the values a , b , c , and d , as follows:

- a - number of pairs (i_1, i_2) where i_1 and i_2 are assigned to the *same* cluster in C_1 and in C_2 . This is a case of agreement.
- b - number of pairs (i_1, i_2) where i_1 and i_2 are assigned to the *same* cluster in C_1 and to *different* clusters in C_2 . This is a case of disagreement.
- c - number of pairs (i_1, i_2) where i_1 and i_2 are assigned to *different* clusters in C_1 and to the *same* cluster in C_2 . This is a case of disagreement.
- d - number of pairs (i_1, i_2) where i_1 and i_2 are assigned to *different* cluster in C_1 and in C_2 . This is a case of agreement.
- n - total number of pairs ($n = a + b + c + d = \frac{m(m-1)}{2}$)

Using this definition of a , b , c , and d , we can construct a contingency table similar to Table 1 for pairs of items, and compute Cohen's Kappa based on this table, which is equivalent to Adjusted Rand Index [31].

4. EMPIRICAL SETTINGS

4.1 The Learning Environment

We use data from an ITS that teaches Fractions for 4th grade. The students progress through the ITS on their own pace, in a linear order defined by the content experts. The subject matter knowledge that the ITS covers is modeled by a Knowledge Graph, which is described in Subsection 4.2 (since it is hierarchical, hereafter we use the term Knowledge Tree).

The content of the ITS includes 550 items, instructional materials such as videos, and on-line labs that students can use to explore the various concepts. These are arranged in 112 learning units. Each of the learning units contains a collection of items and learning materials and is designed to take approximately 5 – 15 minutes.

The course is divided into two parts. Part A contains 57 learning units which include 337 items, and Part B contains 55 learning units which include 213 items. Concepts are first introduced and explained and are later re-visited. This means that items which require a certain skill can appear in different locations.

4.2 Knowledge Tree and Content Mapping

The course was designed according to a Knowledge Tree (KT) that models the hierarchy of skills that students should master (under the root topic "Fractions for 4th grade"). The KT was developed by the content experts who built the course. The first level, termed 'subject', includes 8 topics. Some of these topics have a second level, termed 'sub-subject'. On this level of the tree (sub-subject + subjects

that do not have second level) there are 19 topics. The division of the first two levels is curricular (e.g., ‘adding fractions’ as the first level, with ‘adding fractions with a common denominator’ and ‘adding fractions with a different denominator’ as its children).

In addition to these two levels, there is a third level, termed ‘goals’, which is orthogonal to the classification into subjects and sub-subjects, and refers to the cognitive type of the task (inspired by Bloom’s Taxonomy [1]). Since the ‘goal’ level is orthogonal to the division into subjects/sub-subjects (see Figure 1), it can be interpreted as refining the categories under ‘subject’ (hereafter denoted *first level + goals*), or as refining the ‘sub-subject’ (hereafter denoted *second level + goals*).

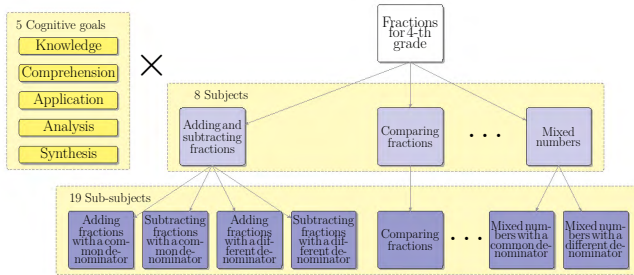


Figure 1: Content expert’s Knowledge tree for the topic “Fractions for 4th grade”. The refining of subjects/sub-subjects into ‘first level + goal’/‘second level + goal’ is computed as a Cartesian product of goals layer with subjects/sub-subjects layers correspondingly.

The experts tagged each item with the ‘subject’, ‘sub-subject’, and ‘goal’ it belongs to. In most cases, each item is mapped into one category on each level. In the few cases where an item was mapped into more than one category, we assume that each unique combination of subjects/sub-subjects is actually a new knowledge component (similar to the rationale of [15]). For example, if item i is marked as belonging to subjects 1 and 2, we create a new artificial subject for this combination of subjects. We removed from the data artificial combinations containing only one item, and the few items (< 5) that belong to these combinations.

4.3 Knowledge Components

We interpret Knowledge Component (KC) as a group of items that deal with the same concept (i.e., require the same skill)³. We examine classifications of items into Knowledge Components that are based on different levels of granularity with respect to the Knowledge Tree. For example, ‘first level’ is a classification that is based only on the first split of the tree (‘subject’). Table 2 presents the number of KCs defined by each level of the KT.

4.4 Data

The data include the responses of 594 4th grade students, who used the ITS for a few hours a week during regular class hours, for a period of 2 months. (We remove the data

³We use the term KC in two ways – as *skill*, and as a *set of items* that require a certain skill

Table 2: Number of Knowledge Components by the level of Knowledge Tree.

Level of Knowledge Tree	Number of Knowledge Components
First	14
First with Goals	42
Second	32
Second with Goals	62

of students who attempted less than 50 items, and the few who had less than 25% success on first attempt, as we assume they were mainly ‘gaming the system’). On average, students spent about 12 hours in the ITS.

Students’ performance is operationalized as *correct on first attempt*. From the log files, we build a 0/1 *student* \times *item* response matrix, denoted RM . $RM[i, j] = 1$ iff students i solved item j correctly on first attempt.

5. RESULTS ON REAL DATA

5.1 Computing the Similarity Matrix

We compute the similarity matrix for each of the four measures, as described in Section 3. This yields four similarity matrices.

To cluster the items based on these matrices, we use three clustering algorithms:

- Ward’s Hierarchical clustering using Pearson correlation Distance
- Ward’s Hierarchical clustering using Euclidean Distance
- K-means clustering using Euclidean Distance

As noted before, the number of clusters is defined according to the number of Knowledge Components of the Knowledge Tree (Table 2). Goodness-of-fit of clustering is evaluated by measuring its similarity to the ground truth labeling, using Adjusted Rand Index (ARI).

5.2 Results of Hierarchical Clustering

Table 3 demonstrates the results of the Hierarchical Clustering on the entire course, based on the four similarity measures, using Pearson Distance (which outperforms Euclidean Distance in all combinations; thus we omit the results for Euclidean Distance). As can be seen, clustering that is based on Kappa Learning outperforms the other measures in all the combinations.

5.3 Results of K-Means Clustering

In addition to the comparison that is based on Hierarchical Clustering, we make a comparison that is based on K-Means Clustering. Since K-Means is non-deterministic (depends on random assignment of initial cluster centers), we run the algorithm 100 times for each combination, each time computing the Adjusted Rand Index against ground truth. The distribution of the Adjusted Rand Index for each combination are presented in Figures 2 and 3. As can be seen, Kappa Learning outperforms the other similarity measures.

Table 3: Adjusted Rand Index for different similarity measures, using Hierarchical Clustering and Pearson Distance, for number of KCs that is based on different levels of the Knowledge Tree.

	First	First with Goals	Second	Second with Goals
Kappa Learning	0.26	0.21	0.26	0.36
Kappa	0.16	0.17	0.18	0.27
Yule	0.15	0.19	0.21	0.29
Pearson	0.16	0.18	0.21	0.30

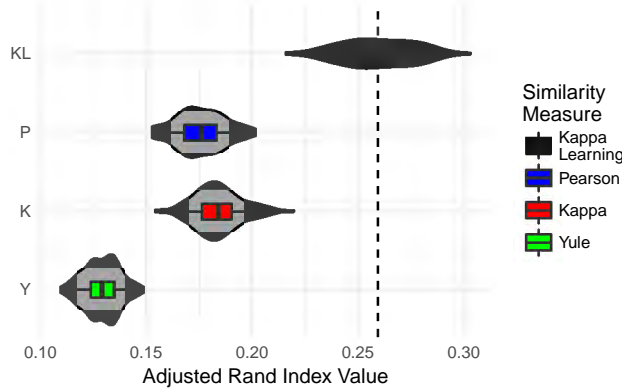


Figure 2: Results of K-means clustering for the entire course and number of KCs as defined by the second level of the Knowledge Tree. The vertical dashed line goes through the mean of the distribution of the ARI results for Kappa Learning.

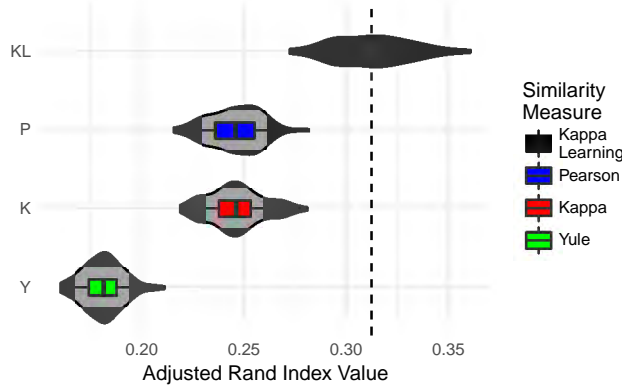


Figure 3: Results of K-means clustering for the entire course and number of KCs as defined by the second level of the Knowledge Tree + Goals. The vertical dashed line goes through the mean of the distribution of the ARI results for Kappa Learning.

5.4 Finding optimal number of clusters

While the clustering algorithms described above take the number of clusters as input, in many real-life scenarios it is unknown in advance and needs to be discovered from the data. Finding an optimal number of clusters is a fundamen-

tal problem in clustering analysis that is typically ill-posed [16], as there is no rigorous definition of a cluster, and the practical considerations are domain and application-specific. For example, in our model, we consider number of clusters that is based on different resolution of experts' hierarchical Knowledge Graph (Subsection 4.2).

The 'goodness' of the resulting clustering is usually measured by cluster cohesion and cluster separation. One of the measures for cluster cohesion or compactness is Within Cluster Sum of Squares (WSS), W_k . For any clustering of a set S into k clusters $S = \{S_1, S_2, \dots, S_k\}$, WSS is defined as

$$W_k = \sum_{i=1}^k \frac{1}{2|S_i|} \sum_{x,y \in S_i} [dist(x,y)]^2 \quad (6)$$

where $dist(x,y)$ is a measure for distance between two items of a set.

In our case the value of W_k depends on the method used for evaluating the item's similarity matrix based on student's performance matrix, the method for measuring the distance between items of similarity matrix and the clustering algorithm used. Within Clusters Sum of Squares is commonly used to find an optimal number of clusters using the 'elbow' heuristic, however, in our case, there is no clear 'elbow' in the graph. Another common method for estimating an optimal number of clusters using WSS measure is the Gap statistic method.

5.4.1 Gap Statistic

The main idea of Gap statistic is comparing the goodness of clustering applied to a specific dataset with the goodness of clustering obtained when applied on a uniformly distributed data with no clustering structure at all (so-called 1-cluster data) [29]. The GAP_k measure used in Gap statistic is the difference between an expected value of $\log(W_k)$ computed for clustering of 1-cluster random data into k clusters and $\log(W_k)$ value obtained from clustering of input dataset into the same number of clusters k . The random data is generated from a uniform distribution over the same range as the input dataset. The Gap statistic method receives $K.max$ – the maximal number of clusters to consider, a clustering algorithm, a distance measure, and an input dataset. For each k from 1 to $K.max$, it computes GAP_k value and searches for the value of k that maximizes the Gap value.

For the four similarity matrices obtained (Section 3) we compute Gap statistic using Ward's Hierarchical Clustering, Pearson distance and $K.max = 70$. Then we apply two different methods for computing the optimal number of clusters: First SE Max (first local maximum of Gap value within one standard error) and First Max (first local maximum of Gap value) [29].

Running Gap statistic on Kappa Learning dataset produces 19 as an optimal number of clusters (see Figure 4), which is similar to the number of Knowledge Components at the Second level of the Knowledge Tree (see Figure 1; the second level of the Knowledge Tree is denoted 'sub-subjects').

The optimal number of clusters based on Yule similarity measure is 14 (with First SE Max) or 15 (with First Max),

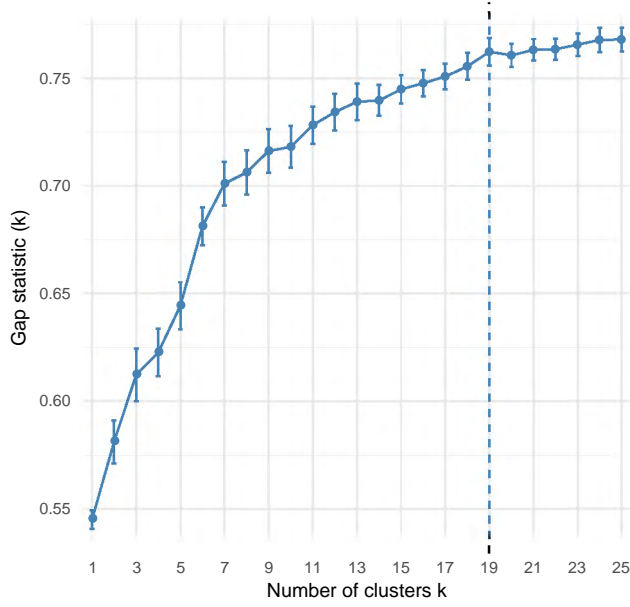


Figure 4: The values of GAP statistic for k in a range from 1 to 25 computed based on Kappa Learning similarity matrix. The vertical dashed line indicates the optimal number of clusters as predicted by both firstMax and firstSEMax methods.

which is also quite close to the ground truth. For the two other methods (Cohen’s Kappa, Pearson), Gap statistic does not produce meaningful results (Table 4).

Table 4: Optimal number of clusters by GAP statistic.

	First Max Method	First SE Max Method
Kappa Learning	19	19
Kappa	1	1
Yule	15	14
Pearson	1	1

6. SIMULATION STUDY

In addition to evaluating our method on data from a real learning environment (Subsection 4.1), we conduct a simulation study.

6.1 Data Generation

Our simulation model makes the following assumptions:

1. Each item belongs to one of K knowledge components (KCs); the items are uniformly distributed among these KCs. Each KC has an individual difficulty level (drawn from a probability distribution defined below).
2. The order of appearance of KCs is predefined. We assume that the topic is first presented and explained to the learners, so the majority ($\approx 60\%$, chosen empirically based on the data) of the items that belong to it appear one after the other. The rest of the items

that belong to the KC are presented to the learner on a later stage and interleaved between items from other KCs.

3. Students learn as they interact with the items; Learners have individual learning rate (drawn from a probability distribution defined below).

6.1.1 Hidden Markov Model and Bayesian Knowledge Tracing

Bayesian Knowledge Tracing (BKT) [10] is a popular approach to model skill acquisition in ITSs. It models a student knowledge as a latent binary variable of a Hidden Markov Model. Learning is modeled as a transition from ‘not mastered’ to ‘mastered’ state. The standard BKT model uses the same four parameters for all the students and items. Several studies extended the basic BKT model with individualized parameters for student ability and item difficulty (e.g., [18, 23, 33]). We use the model introduced in [33] as the underlying model for the data generation process.

6.1.2 Individualized Bayesian Knowledge Tracing

We apply Individualized BKT approach with parameter splitting [33] to model a learning process. Namely, we construct individual HMM per student and KC. All items of the same KC are assumed to have the same difficulty. The model assumes students learn as they practice more. On each opportunity to solve an item that belongs to a knowledge component, the probability that the student masters the skill underlying the item’s KC increases.

Let us define:

- L - number of learners
- K - number of Knowledge Components
- N - total number items (questions)

For each KC k and each student l we generate the following parameters:

- $P(L_0)$ - the probability that a student initially knows a particular KC. In this model we assume the students have no initial knowledge.
- $P(T)_l^k$ - the probability of learning for student l and skill k .
- $P(S)$ - the probability of slip, meaning making an incorrect attempt when applying a known skill. We assume $P(S) = 0.1$ (not individualized; determined by an educational expert).
- $P(G)$ - the probability of random guess, meaning making a correct attempt when applying an unknown skill. We assume $P(G) = 0.2$ (not individualized; determined by an educational expert).

As proposed in [33], the value of the parameters $P(T)_l^k$ is combined from two components: a per-skill component and a per-student component. So, we generate for each skill and

each student a pair of parameters $(P(T)_l, P(T)^k)$. Each of the above parameters is generated from a uniform distribution $\mathcal{U}(0, 1)$. Then for each student l and KC k the parameters are combined as follows:

$$P(T)_i^k = \sigma(l(P(T)_l) + l(P(T)^k)) \quad (7)$$

where:

$$\sigma(x) = 1/(1 + \exp^{-x}) \quad (8)$$

and

$$l(x) = \log(x/1 - x) \quad (9)$$

Where $\sigma(x)$ and $l(x)$ are the sigmoid and logit functions, respectively.

The performance matrix for each student and knowledge concept is generated using R's HMM package, and the data is combined into a $L \times N$ student's performance matrix. For all Knowledge Components containing more than 6 items, the first 6 items are placed one after another, modeling the introduction of the concept to the learners. The rest of the items are shuffled randomly between future KCs.

6.1.3 Model Parameters

In the experiment reported below the basic setting is 1000 learners, 20 Knowledge Components, 200 items. The parameters are chosen in a way that approximates the multivariate distribution of the real data with respect to the average number of items per Knowledge Component and the mean performance of students, as illustrated in Table 5.

Table 5: Comparison of simulation model to empirical data.

	Average number of Questions per KC	Average performance of Students
Empirical data	≈ 9	67%
Simulation model	10	64%

To evaluate the clustering that is based on each of the four measures, we follow the same process as described in Section 3. Since the results depend on the simulated data, we repeat the process 700 times, each time starting with generating a new performance matrix.

6.2 Results on Simulated Data

The results are presented in Table 6 (right column) and Figure 5. As can be seen, Kappa Learning outperforms all other measures in its ability to reproduce the original clusters. Table 6 also presents the results of each measure on the real data (left column), for reference.

To verify the statistical significance of the results, we conduct a t-test for the results of Kappa Learning vs. the three other measures (Yule, Cohen's Kappa, and Pearson). For all combinations, the p-value is less than 0.01.

7. DISCUSSION

The results show that Kappa Learning - the new similarity measure that we propose, which is based on adjusting

Table 6: Comparison of Adjusted Rand Index values for different similarity measures.

	Real data, Second level with Goals	Simulation Model (averaged over 700 runs)
Kappa Learning	0.36	0.40
Kappa	0.27	0.35
Yule	0.29	0.31
Pearson	0.30	0.37

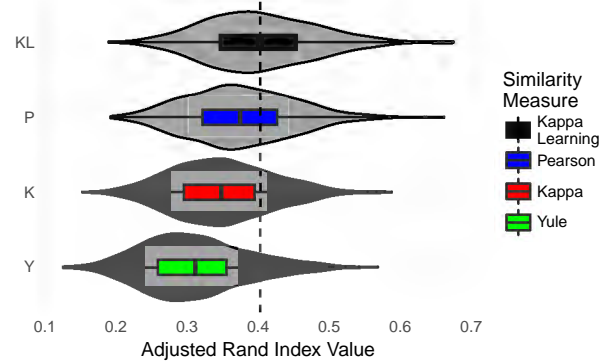


Figure 5: Distribution of Adjusted Rand Index values for different similarity measures (KL - Kappa Learning, K - Kappa, Y - Yule, P - Pearson). The vertical dashed line goes through the mean of the distribution of ARI values for Kappa Learning.

Cohen's Kappa to 'learning', can improve the clustering of educational items into Knowledge Components, compared to the state-of-the-art (the measures that are reported in [26] as producing the best results). We ascribe this to the fact that Kappa Learning explicitly models similarity under the assumption that students' skill can grow during the activity (= learning), while the conventional measures are based on the assumption that students' skill is fixed.

On real data, with different combinations for the number of clusters (Table 2), the improvement with Hierarchical Clustering was in the range of 10 – 60% (Table 3), comparing to the conventional measures (Kappa, Yule, and Pearson). On simulated data that follow the 'mastery' assumption, and allow items of different Knowledge Components to interleave (which makes the task more difficult; if all the items of a certain KC are presented together, the clustering is almost trivial), the improvement with Hierarchical Clustering was in the range of 10 – 20% (Table 6).

In real-life scenarios, the number of clusters, which the clustering algorithms that we use take as input, is typically unknown, and it is necessary to extract it from the data. On the task of finding an optimal number of clusters, Gap statistic on clustering that is based on Kappa Learning yielded a number of clusters (19) that is similar to the number of clusters in the ground truth (according to the second level of the Knowledge Graph. See Table 2). Among the other measures, Gap statistic on Yule-based clustering also produced results that are reasonably close to the ground truth. For

Kappa and Pearson, Gap statistic did not yield meaningful results.

Overall, Kappa Learning was superior with respect to all the combinations that were evaluated: Various interpretations of the ground truth (deciding on the KCs according to different levels of Knowledge Graph); clustering algorithm – K-Means and Hierarchical Clustering; real and simulated data; and in reproducing the number of clusters with Gap statistic. Thus, we conclude that in the context of learning in structured domains (such as K-6 Math), Kappa Learning provides a significant improvement to the task of clustering that is based on item similarity, compared to conventional item-similarity measures.

7.1 Generalizing to Random Order of Items

In Subsection 2.2, the definition of Kappa Learning was based on the assumption that the items are presented to the learners in a fixed order. We now explain how this assumption can be removed. Let us assume that the items administered to the learners in a random order, meaning that different learners may see the items in a different order. In this case, for each learner and for each pair of items we construct the contingency table (similar to Table 1) by computing the values of a , b , c , d as follows:

- a - number of learners who got both items correct
- b - number of learners who got the first item presented to them (among Q_1 and Q_2) correct, and the second incorrect
- c - number of learners who got the first item presented to them (among Q_1 and Q_2) incorrect, and the second correct
- d - number of learners who got both items incorrect

7.2 Future Work

This work provides a few directions for future research. On the next step, we intend to work with the developers of the ITS on using the results of Kappa Learning to refine and optimize the pedagogic design of the ITS (cognitive modeling, but also questions such as which KCs require more content, are too difficult, too easy, etc.).

Algorithmic directions include studying additional ways to insert the notion of ‘learning’ into existing item-to-skill detection methods, and additional sources of information such as domain experts or analysis of the body of the items (text, images, mathematical symbols, etc.).

In terms of use cases, it would be interesting to evaluate Kappa Learning on data from a variety of learning environments (e.g., MOOCs) and subject matters, and in particular, on domains in which knowledge is less structured (e.g., reading comprehension).

7.3 Summary and Conclusions

This paper presents a new method for measuring the similarity between educational items, termed Kappa Learning. The novelty of this method, compared to previous measures of similarity between educational items, lies in the fact that

it explicitly captures the notion of ‘learning’, namely, change of the latent trait (student’s mastery of the concept). This is done by extending the notion of ‘agreement’ within Cohen’s Kappa basic formula.

Our results show that clustering that is based on Kappa Learning outperforms clustering that is based on conventional methods (Cohen’s Kappa, Yule, Pearson), on real data from K-6 Math ITS that teaches multiple concepts, and on generated data that simulates learning of multiple, interleaved concepts. Thus, we believe that Kappa Learning is more suitable than existing measures for computing similarity between items in the context of learning in structured domains.

8. ACKNOWLEDGMENTS

This research was supported by The Willner Family Leadership Institute for the Weizmann Institute of Science, Iancovich-Fallmann Memorial Fund, established by Ruth and Henry Yancovich, and by Ullmann Family Foundation. The authors would like to thank The Center for Educational Technology for giving us access to the data, and to Hagar Rubinek and Yael Weisblatt for their help in conducting this research. The authors would like to thank Ido Roll for useful comments.

9. REFERENCES

- [1] L. W. Anderson, D. R. Krathwohl, P. W. Airasian, K. A. Cruikshank, R. E. Mayer, P. R. Pintrich, J. Rath, and M. C. Wittrock. A taxonomy for learning, teaching, and assessing: A revision of bloom’s taxonomy of educational objectives, abridged edition. *White Plains, NY: Longman*, 2001.
- [2] T. Balint, R. Teodorescu, K. Colvin, Y.-J. Choi, and D. E. Pritchard. Identifying characteristics of pairs of questions that students answer similarly. In *Physics Education Research Conference 2015*, pages 55–58, 2015.
- [3] T. Barnes. The q-matrix method: Mining student response data for knowledge. In *American Association for Artificial Intelligence 2005 Educational Data Mining Workshop*, pages 1–8, 2005.
- [4] B. S. Bloom. Learning for mastery. *Instruction and Curriculum*, 1968.
- [5] P. Boroš, J. Nižnan, R. Pelánek, and J. Řihák. Automatic detection of concepts from problem solving times. In H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Artificial Intelligence in Education*, pages 595–598, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
- [6] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis – a general method for cognitive model evaluation and improvement. In M. Ikeda, K. D. Ashley, and T.-W. Chan, editors, *Intelligent Tutoring Systems*, pages 164–175, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
- [7] S.-s. Choi and S.-h. Cha. A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, pages 43–48, 2010.
- [8] J. Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

- [9] A. T. Corbett and J. R. Anderson. Student modeling and mastery learning in a computer-based programming tutor. In C. Frasson, G. Gauthier, and G. I. McCalla, editors, *Intelligent Tutoring Systems*, pages 413–420, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.
- [10] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [11] J. de la Torre. An empirically based method of q-matrix validation for the dina model: Development and applications. *Journal of Educational Measurement*, 45(4):343–362, 2008.
- [12] M. C. Desmarais, B. Beheshti, and R. Naceur. Item to skills mapping: Deriving a conjunctive q-matrix from data. In S. A. Cerri, W. J. Clancey, G. Papadourakis, and K. Panourgia, editors, *Intelligent Tutoring Systems*, pages 454–463, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- [13] M. C. Desmarais and M. Gagnon. Bayesian student models based on item to item knowledge structures. In W. Nejdl and K. Tochtermann, editors, *Innovative Approaches for Learning and Knowledge Sharing*, pages 111–124, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
- [14] M. C. Desmarais and R. Naceur. A matrix factorization method for mapping items to skills and for enhancing expert-based q-matrices. In H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Artificial Intelligence in Education*, pages 441–450, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
- [15] Y. Huang, J. P. González-Brenes, and P. Brusilovsky. General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge. In *Proceedings of the 7th International Conference on Educational Data Mining, EDM 2014*, pages 84–91, 2014.
- [16] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.
- [17] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.
- [18] M. Khajah, R. Wing, R. Lindsey, and M. Mozer. Integrating latent-factor and knowledge-tracing models to predict individual differences in learning. In *Proceedings of the 7th International Conference on Educational Data Mining, EDM 2014*, pages 99–106, 2014.
- [19] K. R. Koedinger, E. A. McLaughlin, and J. C. Stamper. Automated student model improvement. In *Proceedings of the 5th International Conference on Educational Data Mining, EDM 2012*, pages 17–24, 2012.
- [20] S. Lee, Z. Chen, D. Pritchard, A. Kimn, and A. Paul. Factor analysis reveals student thinking using the mechanics reasoning inventory. In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale, L@S '17*, pages 197–200, 2017.
- [21] J. Liu, G. Xu, and Z. Ying. Data-driven learning of q-matrix. *Applied Psychological Measurement*, 36(7):548–564, 2012.
- [22] R. Liu and K. R. Koedinger. Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains. *Journal of Educational Data Mining*, 9(1):25–41, 2017.
- [23] Z. A. Pardos and N. T. Heffernan. Kt-idem: introducing item difficulty to the knowledge tracing model. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 243–254. Springer, 2011.
- [24] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis –a new alternative to knowledge tracing. In *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*, pages 531–538. IOS Press, 2009.
- [25] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.
- [26] J. Rihak and R. Pelanek. Measuring Similarity of Educational Items Using Data on Learners’ Performance. In *Proceedings of the 10th International Conference on Educational Data Mining, EDM 2017*, pages 16–23, 2017.
- [27] J. C. Stamper and K. R. Koedinger. Human-machine student model discovery and improvement using datashop. In G. Biswas, S. Bull, J. Kay, and A. Mitrovic, editors, *Artificial Intelligence in Education*, pages 353–360, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
- [28] K. K. Tatsuoaka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20(4):345–354, 1983.
- [29] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.
- [30] M. J. Warrens. On association coefficients for 2×2 tables and properties that do not depend on the marginal distributions. *Psychometrika*, 73(4):777, 2008.
- [31] M. J. Warrens. On the equivalence of cohen’s kappa and the hubert-arabie adjusted rand index. *Journal of Classification*, 25(2):177–183, 2008.
- [32] Y. Xu and J. Mostow. Comparison of methods to trace multiple subskills: Is lr-dbn best? In *Proceedings of the 5th International Conference on Educational Data Mining, EDM 2012*, pages 41–48, 2012.
- [33] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International Conference on Artificial Intelligence in Education*, pages 171–180, 2013.

Using Knowledge Component Modeling to Increase Domain Understanding in a Digital Learning Game

Huy Nguyen
Carnegie Mellon University
hn1@andrew.cmu.edu

Yeyu Wang
University of Pennsylvania
yeyuw@upenn.edu

John Stamper
Carnegie Mellon University
jstamper@cs.cmu.edu

Bruce M. McLaren
Carnegie Mellon University
bmclaren@cs.cmu.edu

ABSTRACT

Knowledge components (KCs) define the underlying skill model of intelligent educational software, and they are critical to understanding and improving the efficacy of learning technology. In this research, we show how learning curve analysis is used to fit a KC model - one that was created after use of the learning technology - which can then be improved by human-centered data science methods. We analyzed data from 417 middle-school students who used a digital learning game to learn decimal numbers and decimal operations. Our initial results showed that problem types (e.g., ordering decimals, adding decimals) capture students' performance better than underlying decimal misconceptions (e.g., longer decimals are larger). Through a process of KC model refinement and domain knowledge interpretation, we were able to identify the difficulties that students faced in learning decimals. Based on this result, we present an instructional redesign proposal for our digital learning game and outline a framework for post-hoc KC modeling in a tutoring system. More generally, the method we used in this work can help guide changes to the type, content and order of problems in educational software.

Keywords

KC Model, Decimal Number, Digital Learning Game

1. INTRODUCTION

In the view of KC modeling, student's knowledge can be treated as a set of inter-related KCs, where each KC is "an acquired unit of cognitive function or structure that can be inferred from performance on a set of related tasks" [22]. A KC-based student model (which we refer to as *KC model*) has been employed in a wide range of learning tasks, such as supporting individualized problem selection [11], choos-

ing examples for analogical comparison [35] and transitioning from worked examples to problem solving [43]. A good KC model is vital to intelligent educational software, particularly in the design of adaptive feedback, assessment of student knowledge and prediction of learning outcomes [24].

A new area in educational technology that could potentially benefit from KC models is digital learning game. While there has been much enthusiasm about the potential of digital games to engage students and enhance learning, few rigorous studies have demonstrated their benefits over more traditional instructional approaches [32, 34]. One possible reason is that most digital learning games have been designed in a one-size-fits-all approach rather than with personalized instruction in mind [9]. Adopting KC modeling techniques could therefore be an important first step in meeting individual students' learning needs and making digital learning games a more effective form of instruction. A critical question in this direction is whether a KC model can be created after the use of the learning technology, in order to better understand the targeted learning domain and to help in improving the technology.

In our study, we explore this question in the context of a game that teaches decimal numbers and decimal operations to middle-school students. We started with an initial KC model based on problem type (e.g., adding decimals, completing sequences of decimals), then used the human-machine discovery method [51] to derive new KCs and formulate the best fitting model. From this improved model, we first discuss findings about students' learning of decimal numbers and propose potential changes to the instructional materials that address a wider range of learning difficulties - a process known as "closing the loop" [24]. Then, we outline a general framework for adding KC models to educational software in a post-hoc manner and discuss its broader implications in digital learning games.

2. BACKGROUND

In this section, we first present background information about two aspects of student modeling that are relevant to our work: (1) KC modeling, a technique that represents students' knowledge as latent variables, and (2) the current state of student modeling in digital learning games. Then,

Huy Nguyen, Yeyu Wang, John Stamper and Bruce McLaren "Using Knowledge Component Modeling to Increase Domain Understanding in a Digital Learning Game" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 139 - 148

we describe the game environment used for data analysis.

2.1 KC Modeling

Traditionally, KC models have been developed by domain experts, using Cognitive Task Analysis methods such as structured interviews, think aloud protocols and rational analysis [45]. These methods result in better instructional design but are also highly subjective and require substantial human effort. To address this shortcoming, a wide range of prior research has focused on creating KC models through data-driven techniques. Some of the earliest work on identifying and improving KC models was done by Corbett and Anderson [11] with the early LISP tutors. In this work, plotting of learning curves showed “blips” or “peaks” in the curves which indicated new KCs that were not accounted for in the initial model. By using a computational model to fit the data in learning curves, [5] showed how Learning Factor Analysis (LFA) could automate the process of identifying additional KCs in educational software. LFA takes as input a space of hypothesized KCs, which can be discovered through visualization and analysis tools [51]. Once there are several human-generated KC models, they can be combined by merging and splitting skills using machine learning techniques that aim to improve the overall fit [23].

It is important to define a good model, but it is not always clear how to do so. Goodness of fit is best measured by cross validation, but this technique is time consuming and computationally expensive for large datasets. Furthermore, there is no consensus on how cross validation should be performed on educational data [50]. Two related and easy-to-compute metrics are the Akaike information criterion (AIC) and Bayesian information criterion (BIC), which address overfit by measuring prediction accuracy while penalizing complexity. In general, a lower AIC/BIC/cross validation score indicates a better model. In case they do not agree, [50] showed that AIC correlates with cross validation better than BIC, through an analysis of 1,943 KC models in DataShop. However, these scores alone do not portray the full picture; as pointed out by [3], many student modeling techniques that aim to predict student learning achieve negligible accuracy gains, “with differences in the thousandths place,” suggesting that they are already close to ceiling performance. In response, [28] brought attention to another important criterion - whether the model is *interpretable* and *actionable*. As the authors argued, even slight improvement can be meaningful if it reveals insights on student learning that generalize to a new context and lead to better, empirically validated instructional designs. For instance, some research has been successful in redesigning tutor units to help students reach mastery more efficiently, based on analysis of previous KC models [24, 27].

Our analysis follows the established process outlined above, in which we started with a basic human-generated KC model, then identified potential improvements using learning curve analysis, and evaluated the new model by AIC, BIC and cross validation. We also derived instructional insights from this model as the first step in closing the loop.

2.2 Student Modeling in Games

As pointed out by [2], knowledge in digital learning games is harder to represent than knowledge in tutoring systems

because the students’ thinking process, as well as learning objectives, may not be as explicit. The popular student modeling techniques for learning games are those that can represent uncertainty, such as Bayesian Networks (BN) [31] and Dynamic Bayesian Networks (DBN) [8]. For instance, in *Use Your Brainz*, by applying BN to each level of the game to estimate the problem-solving skills of learners, researchers were able to validate their measures of stealth assessment [46]. [10] applied DBN in *Prime Climb*, a math game for learning factorization, to build an intelligent pedagogical agent that results in more learning gains for students. Follow-up work by [30] refined and evaluated the existing DBN, yielding substantial improvement in the model’s test performance prediction accuracy, which in turn helps better estimate students’ learning states in future studies. As another example, [42] employed DBN to predict responses on post-test questions in *Crystal Island*, an immersive narrative-based environment for learning microbiology.

Recent research has proposed entirely data-driven methods for discovering KC models in a tutoring system [17, 26]. However, most KC models employed in digital learning games have been generated manually by domain experts. For instance, in *Zombie Division*, the KCs were identified by math teachers as common prime factors such as “divide by two” and “divide by three” [2]. Similarly, the designers of *Crystal Island* labeled the general categories of knowledge involved in problem-solving as *narrative*, *strategic*, *scenario solution* and *content* knowledge [42]. The first attempt to refine a human-generated baseline KC model using data-driven techniques in digital learning games was done by Harpstead and Alevan [18]. Their approach, which was applied to *Beanstalk*, a game that teaches the concept of physical balance, is based on [51]’s human-machine discovery method, which is very similar to ours; however, there are notable differences in the learning environments. In particular, the domain of decimal numbers involves many more rules and operations than *Beanstalk*’s domain of beam balancing; in turn, our digital learning game also incorporates more activities (e.g., placing numbers on a number line, completing sequences, assigning numbers to less-than and greater-than buckets). Therefore, our KC modeling process takes into account not just the instructional materials but also elements of the interface and problem types, which could be more generalizable to other learning environments.

2.3 A Digital Learning Game for Decimals

Decimal Point is a single-player game that helps middle-school students learn about decimal numbers and their operations (e.g., adding, ordering, comparing). The game is based on an amusement park metaphor (Figure 1), where students travel to various areas of the park, each with a different theme (e.g., *Haunted House*, *Sports World*), and play a variety of mini-games within each theme area, each targeting a common decimal misconception: **Megz** (longer decimals are larger), **Segz** (shorter decimals are larger), **Pegz** (the two sides of a decimal number are separate and independent) and **Negz** (decimals smaller than 1 are treated as negative numbers) [21, 47]. Each mini-game also involves one of the following *problem types*:

1. **NumberLine** - locate the position of a given decimal number on the number line.

2. **Addition** - add two decimal numbers by entering the carry digits and the result digits.
3. **Sequence** - fill in the next two numbers of a given sequence of decimal numbers.
4. **Bucket** - compare given decimal numbers to a threshold number and place each decimal in a “less than” or “greater than” bucket.
5. **Sorting** - sort a given list of decimal numbers in ascending or descending order.



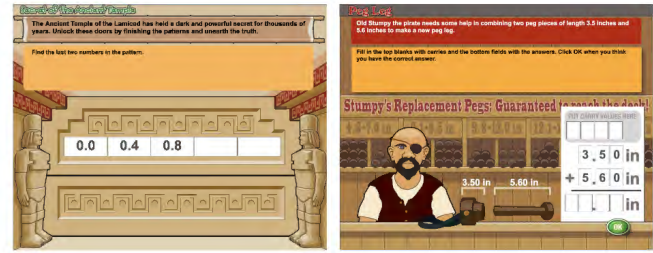
Figure 1: A screenshot of the main map screen.

In each theme area, and across the different theme areas, the problem types are interleaved to improve mathematics learning [41] and introduce variety and interest in gameplay. Figure 2 shows the screenshots of two mini-games - *Ancient Temple* (a **Sequence** game) and *Peg Leg Shop* (an **Addition** game). Each mini-game requires students to solve up to three problems of the same type (e.g., place three numbers on a number line, or complete three number sequences). Students must answer correctly to move to the next mini-game; they also receive immediate feedback about their answers. To further support learning, after a problem has been solved, students are prompted to self-explain their answer by selecting from a multiple-choice list of possible explanations [7].

A prior study by [34] showed that *Decimal Point* promoted more learning and enjoyment than a conventional instructional system with identical decimal content. Follow-up studies by [37] and [19] then tested the effect of student agency, where students can choose the order and number of mini-games they play. These studies revealed no differences in learning or enjoyment between low- and high-agency conditions, but [19] found that students in a high-agency condition had the same learning gains while playing fewer mini-games than those in low-agency, suggesting that the high-agency version led to more learning efficiency.

Post-hoc analyses by [52] examined the different mini-game sequences played by high-agency students and found that, consistent with the reports in [19], those who stopped early learned as much as those who played all mini-games. This result leads to important questions about the right amount of instructional content to maximize learning efficiency. To answer these questions, we would need a more fine-grained

measure of student learning using in-game data rather than external test scores. The KC modeling work presented here represents the first step in this direction.



(a) Ancient Temple (b) Peg Leg Shop

Figure 2: Screenshots of two mini-games.

2.3.1 Participants and Design

We obtained data from two prior studies of *Decimal Point* involving 484 students in 5th and 6th grade, in all study conditions [19, 37], and removed those students who did not finish all of the required materials, reducing the sample to 417 students (200 males, 216 females, 1 declined to respond). The students played either some or all of the 24 mini-games in Figure 1, depending on their assigned agency condition, as described previously. When selecting a mini-game, students would play two instances of that game, with the same interface and game mechanics but different questions. Students in the high-agency condition also had the choice to play a third instance of each mini-game once. In subsequent analyses, we use an index of 1, 2 and 3 to denote the instance number, e.g., *Ancient Temple 1*, *Ancient Temple 2* and *Ancient Temple 3*. For a detailed description of the experimental design of prior studies, refer to [19, 37].

2.4 Dataset

We analyzed students’ in-game performance data, which was archived in the DataShop repository [49] in dataset number 2906. The dataset covers a total of 613,055 individual transactions, which represent actions taken in the mini-games by 417 students in solving decimal problems.

3. METHODS & RESULTS

We started with the baseline KC models derived from two sets of features that *Decimal Point* was built upon. These initial models were fit using the Additive Factors Model (AFM) method [6], and the learning curves were visualized in DataShop. AFM is a specific instance of logistic regression, with student-correctness (0 or 1) as the dependent variable and with independent variable terms for each student, each KC, and the KC by opportunity interaction. It is a generalization of the log-linear test model [54] produced by adding the KC by opportunity terms. We then chose the model with better fit and analyzed its learning curves. Each model was run on 42,637 observations tagged with KCs.

3.1 Baseline Models

Our first baseline model, called *DecimalMisc*, consists of four KCs that are the misconceptions targeted by the mini-games: *Megz*, *Segz*, *Negz*, *Pegz* [21]. Because each mini-game was designed based on a single misconception (KC),

we created a model that maps each mini-game question to its corresponding KC. The second model, **ProblemType**, instead maps each mini-game question to its problem type (one of **NumberLine**, **Addition**, **Bucket**, **Sorting** and **Sequence**). Table 1 shows the fit statistics results of these two models.

Table 1: Fit statistics results of the two baseline models. RMSE indicates 10-fold cross-validation root mean squared error, stratified by item. Values that indicate best fit are in bold.

Model (# of KCs)	AIC	BIC	RMSE
DecimalMisc (4)	30,699.27	34,379.97	0.3292
ProblemType (5)	29,504.09	33,202.12	0.3231

As can be seen, **ProblemType** outperforms **DecimalMisc** in all three metrics - AIC, BIC and RMSE. In other words, the actual problem types capture students’ learning better than the underlying misconceptions. In subsequent analyses, we therefore focused on improving the **ProblemType** model. The first step is identifying potential improvements in the learning curve of each KC. In general, a good learning curve is smooth and decreasing [51]. Smoothness indicates that no step is much harder or easier than expected, and a decreasing curve shows that students were learning well and therefore made fewer errors at later opportunities [36].

From Figure 3, we observed that the learning curves of **NumberLine** and **Bucket** are reasonably good. The learning curve of **Addition** stays at roughly the same low error rate throughout ($< 10\%$), but there are sudden peaks, suggesting that some problems were harder than others and thus should be represented by a separate KC. The learning curve of **Sequence** decreases but not smoothly; the zigzag pattern indicates that students were alternating between easy and hard problems. Again, having separate KCs for the lows and highs of the curve would likely yield a better fit. The learning curve of **Sorting** is neither decreasing nor smooth; therefore, this KC needs to be further decomposed.

3.2 Improved KC Models

3.2.1 KC decomposition

To find possible decompositions, we followed the human-machine discovery method outlined in [51] and consulted prior literature on students’ learning of decimal numbers. Below we present our analysis of each problem type.

NumberLine. As its learning curve is already good, we turned to related work on the game *Battleship Numberline* [29], where students have to place given fraction numbers on a number line. The authors found that, on a number line that runs from 0 to 1, students have better understanding when adjusting from 0 or 1 (e.g., $1/10$ or $9/10$) than from $1/2$ (e.g., $3/7$). Since decimal numbers can be translated to fractions and vice versa, we (tentatively) experimented with applying the findings of [29] to our model. In particular, we decomposed the **NumberLine** KC into **NumberLineMid** (the number to locate lies between 0.25-0.75) and **NumberLineEnd** (the number to locate lies between 0-0.25 or 0.75-1).

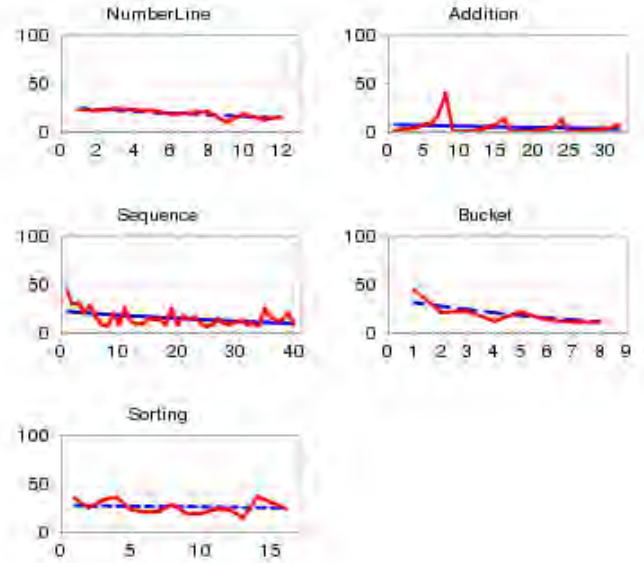


Figure 3: Learning curves of the KCs in ProblemType. The x-axis denotes opportunity number for each KC and y-axis denotes error rate (%). The red line plots all of the actual students’ error rate at each opportunity, while the blue line is the curve fit by AFM.

Addition. There are eight items in an **Addition** game: four text boxes for carry digits - *carryTens*, *carryOnes*, *carryTenths*, *carryHundredths* - and four text boxes for the result - *ansTens*, *ansOnes*, *ansTenths*, *ansHundredths* (see Figure 2b for an example). Previously, all of these items had the same KC label of **Addition**, but we expected that some digits would be harder to compute than others. For instance, the *carryHundredths* digit is always 0, because our problems only involve numbers with two decimal places. On the other hand, because the focus of **Addition** problems is to test that students can carry from the decimal portion to the whole number portion (i.e., probing for the **Pegz** misconception), the *carryOnes* digit is always expected to be 1. It was indeed the case that *carryOnes*, along with *ansOnes*, accounts for a large portion of the peaks in **Addition**’s learning curve (Figure 3). The most common error in these peaks, however, comes from *carryTens* and *ansTens* in the mini-game *Thirsty Vampire 1*. For the majority of students in our sample (87.5%), *Thirsty Vampire 1* was the first **Addition** problem they encountered, and its question ($7.50 + 3.90$) was also the only one with a carry in the tens place; in other words, it was both the first and hardest question. For this reason, we decided to decompose the **Addition** KC into:

- **Addition_Tens_NonZero** applies to the *carryTens* and *ansTens* item in *Thirsty Vampire 1*.
- **Addition_Ones** applies to *carryOnes* and *ansOnes* in all **Addition** mini-games.
- Other items (e.g., *carryTenths*, *carryHundredths*, *ansTenths*) retain the KC label **Addition**.

Sequence. In a **Sequence** mini-game, students have to enter the last two numbers in an increasing arithmetic se-

quence, based on the pattern of the first three given numbers (e.g., Figure 2a). In the way the questions were designed, the first number to fill in always requires an addition with carry, whereas the second does not involve a carry. We therefore hypothesized that the first number is more difficult than the second, which was confirmed by inspection of the learning curve: the alternate up and down patterns depict students' error rates as they filled in the first and second number in each sequence. We further distinguished between numbers with two decimal digits and those with one, as the former should be more difficult to work with. In summary, we decomposed the **Sequence** KC into four KCs: **Sequence_First_OneDigit** (first number, with one decimal digit), **Sequence_First_TwoDigits** (first number, with two decimal digits), **Sequence_Second_OneDigit** (second number, with one decimal digit), **Sequence_Second_TwoDigits** (second number, with two decimal digits).

Bucket. As the learning curve of **Bucket** is already good, we did not further decompose this KC.

Sorting. The learning curve of **Sorting** remains flat at around a 25% error rate. Since there are no outstanding blips or peaks in this curve, we instead used DataShop's Performance Profiler tool to plot the predicted and actual error rates of each mini-game problem (Figure 4). We identified five mini-game problems in which the actual error rate was larger than predicted by at least 5%; in other words, these problems were harder than expected. Therefore, we labeled five of them - *Rocket Science 1*, *Rocket Science 2*, *Jungle Zipline 2*, *Balloon Pop 2* and *Whac A Gopher 1* - by a separate KC called **SortingHard**, while other problems remained in **Sorting**. We will characterize the mathematical features of these **SortingHard** problems in Section 4.2.



Figure 4: Visualization of the **Sorting KC's goodness of fit with respect to ten **Sorting** mini-games with the highest error rates. The bars (shaded from left) show the actual error rates and the blue line shows predicted error rates.**

3.2.2 New model result & comparison

Table 2 shows the fit scores of the original **ProblemType** model, the models resulting from individual KC decompositions, and the final model combining all decompositions, called **Combined**. Apart from **ProblemType** and **Combined**, the name of each other model indicates which original problem type KC is decomposed. For instance, the **Sorting** model has six KCs - **SortingHard**, **Sorting**, **NumberLine**, **Bucket**, **Addition**, **Sequence** - where the last four are identical to those in **ProblemType**. We can therefore see that

decomposing the original **Sorting** KC alone results in a decrease of AIC by 231.91 and BIC by 214.59.

Table 2: Fit statistics results of the original and new models, sorted by AIC in descending order. Values that indicate best fit are in bold.

Model (# of KCs)	AIC	BIC	RMSE
ProblemType (5)	29,504.09	33,202.12	0.3231
NumberLine (6)	29,492.48	33,207.83	0.3233
Sorting (6)	29,272.18	32,987.53	0.3215
Sequence (8)	29,159.27	32,909.25	0.3234
Addition (7)	29,025.77	32,758.43	0.3235
Combined (12)	28,436.07	32,255.34	0.3196

Figure 5 shows the resulting learning curves of the above decompositions. We observed three KCs with issues: (1) **Sequence_First_TwoDigits** is a flat curve which indicates no learning, (2) **SortingHard** remains at high error rates, and (3) **Addition_Tens_NonZero** has too little data (because it only applies to *Thirsty Vampire 1*). Three other KCs - **Addition**, **Addition_Ones**, **Sequence_Second_Digits** - have low and flat curves, suggesting that students already mastered them early on and did not need as much practice (i.e., they were over-practicing with these KCs). The remaining KCs have smooth and decreasing curves. Most notably, we were able to fix the zigzag pattern in the original **Sequence** curve, reduce the peaks in the **Addition** curve, and capture the **Sorting** problems that do reflect students' learning.

Other than **NumberLine**, all of the new models resulted in better AIC and BIC scores. The **Combined** model, which incorporates all decompositions, is the best fit; when compared to **ProblemType**, its AIC score is lower by 1068.02 and its BIC is lower by 946.78. Using DataShop's Performance Profiler tool, we were also able to visualize the differences between these models in Figure 6. Here we see that for each of the new KCs, the **Combined** model's prediction, represented by the blue line (square points), is closer to the actual error rate than the **ProblemType** model's prediction, represented by the green line (round points). Hence, the combination of our KC decompositions resulted in a better fit visually.

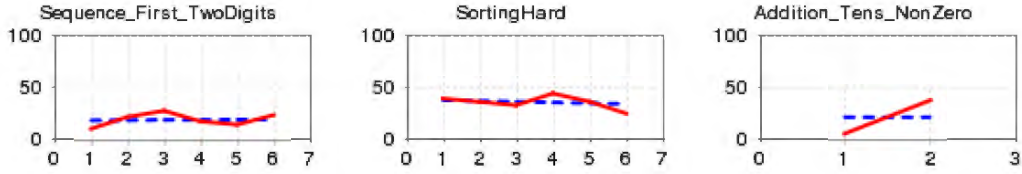
4. DISCUSSION

4.1 Comparison of Baseline Models

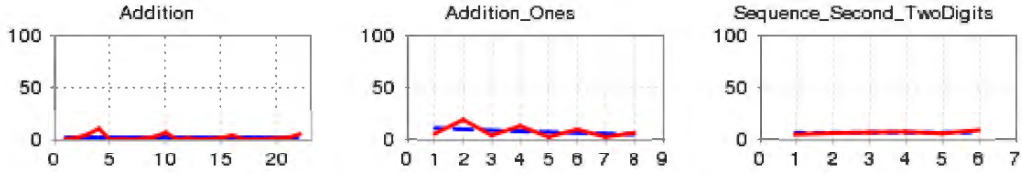
We found that the **ProblemType** model, which maps mini-game questions to problem types, is a better fit for student learning than the **DecimalMisc** model, which maps mini-game questions to underlying misconceptions. Here we outline two possible interpretations.

First, while each question was designed to test one misconception, students may demonstrate other misconceptions in their answers. For example, the mini-game *Jungle Zipline 1*, labeled as **Segz** (shorter decimals are larger), asks students to sort the decimals 1.333, 1.33, 1.3003, 1.3 from smallest to largest. An answer of 1.3003, 1.333, 1.33, 1.3 would match

KCs with issues



Low and flat KCs



Good KCs

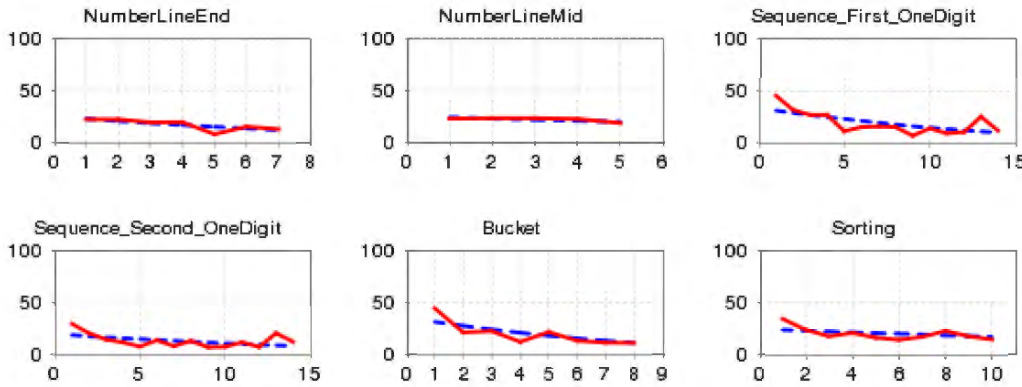


Figure 5: Learning curves of the KCs in Combined. The x-axis denotes opportunity number and y-axis error rate (%). The red line plots the actual students' error rate at each opportunity, while the blue line is the curve fit by AFM.

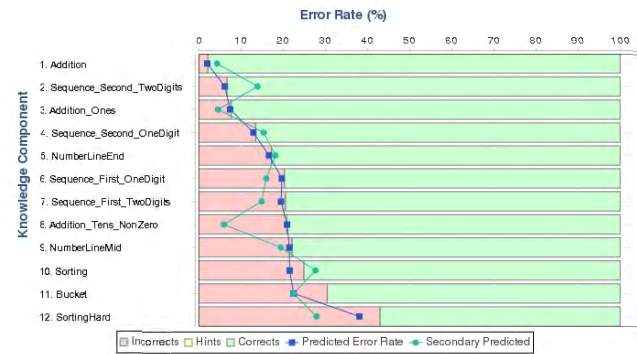


Figure 6: Visualization of the Combined and Problem-Type models' goodness of fit with respect to the new KCs. The bars (shaded from left) show the actual error rates. The blue and green line show predicted error rates of Combined and ProblemType respectively.

the *Segz* misconception, but we observed that 25% of the incorrect answers were 1.3, 1.33, 1.333, 1.3003, which instead corresponds to *Megz* (longer decimals are larger). As another

example, the mini-game *Capture Ghost 1*, labeled as *Megz*, asks students to decide if each of the following numbers - 0.5, 0.341, 0.213, 0.7, 0.123 - is smaller or larger than 0.51. 14% of the incorrect answers stated that $0.5 > 0.51$ and also $0.341 > 0.51$, which demonstrates both *Segz* and *Megz*, respectively. In general, in a problem solving environment like *Decimal Point*, measuring students' misconceptions should be based on their actual answers, not the questions alone. Therefore, a KC model that maps each question to its hypothesized misconception may not capture the students' full range of learning difficulties. Two alternative approaches used by other research for tracking decimal misconceptions are: (1) measuring them at a larger grain size, such as *whole number*, *role of zero* and *fraction* [14], and (2) using erroneous examples instead of problem solving questions [21]. In the context of KC modeling, we could apply our process to an existing dataset of student learning of decimal numbers from erroneous examples, such as the dataset from [33].

From a cognitive perspective, [44] pointed out that "different kinds of knowledge and competencies only show up intertwined in behavior, making it hard to measure them validly and independently of each other." The authors conducted

a series of studies to test students' conceptual knowledge of decimal numbers and procedural knowledge of locating them on a number line. Each study employed four common hypothetical measures of each kind of knowledge, but revealed substantial problems with the measures' validity, suggesting that it is difficult to reliably separate tests of conceptual knowledge and procedural knowledge. In our context, the decimal misconceptions reflect conceptual knowledge while the problem types require a combination of both conceptual and procedural knowledge. Therefore, differentiating problems by their types creates clearer KC distinctions than by their associated misconceptions, because the former matches more closely with students' actual performance.

4.2 Interpretation of the New KCs

Here we discuss the insights from our earlier KC decomposition results, using a combination of learning curve analyses and domain-specific interpretations. While the example questions we cite are specific to those in *Decimal Point*, the findings about student learning are applicable to any other educational technology system in decimal numbers.

NumberLine. Unlike [29], we did not observe that students have more difficulty with numbers close to 0.5 than with numbers close to 0 or 1. Decomposing **NumberLine** into **NumberLineEnd** and **NumberLineMid** results in increases in BIC and RMSE, which are indicative of overfit. Furthermore, the original learning curve of **NumberLine** is already smooth and decreasing (Figure 3), so it is unlikely that any decomposition would yield significant improvements. More generally, this result suggests that students could learn to estimate the magnitude of a given decimal number between 0 and 1 reasonably well, even though they may have difficulty with the equivalent fraction form in the way [29] reported. To explain this difference, we should note that students tend not to perceive decimals and fractions as being equivalent [47], hence difficulties with fractions may not translate to difficulties with decimal numbers. As [12] pointed out, a fraction a/b represents both the relation between a and b and the magnitude of the division of a by b , whereas a decimal number, without the relational structure, more directly expresses a one-dimensional magnitude. Therefore, students often have higher accuracy in estimating decimal numbers than fractions on a number line [53]. The findings from our analysis and [29] further support this distinction.

Addition and Sequence. These problem types both involve computing the sum of two decimal numbers, and as our decompositions showed, the difficulty factor lies in carrying digits to the next highest place value. In the case of **Addition**, the first question, which also happens to be the most challenging, is to add 7.50 and 3.90, which requires two carries, one to the ones place and one to the tens place. The error rate is therefore highest for this question (the first peak in Figure 3), but decreases at later (easier) opportunities. The original learning curve of **Sequence** problems has a zigzag pattern due to the students alternating between additions with and without carry. Distinguishing between these two types of operations, and also on the number of decimal digits, did result in a better model fit. We also note that the error rates in **Sequence** problems are generally higher than in **Addition** problems. A possible interpretation is that, while the underlying addition operations are similar,

the **Sequence** interface does not lay out the carry and result digits in detail as the **Addition** interface does (Figure 2). As pointed out by [25], for adding and subtracting decimals of different lengths, incorrect alignment of decimal operands is the most frequent source of error. Since **Addition** problems already supported this alignment via the interface, students were less likely to make mistakes in them.

Bucket and Sorting. These problem types both involve performing comparisons in a list of five decimal numbers, but in different manners. **Bucket** problems require comparing each number to a given threshold value, while **Sorting** problems require comparing the numbers among themselves. According to [40], ordering more than two decimals (**Sorting**) could reveal latent erroneous thinking which mere comparison of pairs (**Bucket**) cannot. Consistent with this finding, our results also showed that students were able to learn **Bucket** problems well but struggled with **Sorting**. Our hypothesis is that a **Sorting** problem requires two separate skills: (1) comparing individual pairs of number (in a list of five numbers, students may perform up to ten comparisons), and (2) ordering the numbers once all the comparisons have been established. The current interface only asks for the final sorted list, so it would need to be redesigned to allow for tracking student mastery of each of these two skills. Furthermore, by examining the five problems categorized as **SortingHard**, we identified unique challenges that were not present elsewhere in *Decimal Point*. First is the issue of negative number - the mini-game *Balloon Pop 2*, with an error rate close to 60% (Figure 4), asks students to sort the sequence 8.5071, -8.56, 8.5, -8.517 in descending order. Given that students may hold misconceptions about both the length and sign of decimal numbers [21], and that no other **Sorting** problems involve negative numbers, it is clear why students faced significant difficulties in this case. The second issue is another common misconception - that a 0 immediately to the right of the decimal point does not matter (e.g., $0.03 = 0.3$) - which [39] referred to as *role of zero*. It could be invoked in the mini-game *Rocket Science 1*, which asks students to sort 0.14, 0.4, 0.0234, 0.323 in ascending order; in particular, 19% of the incorrect answers put 0.0234 between 0.14 and 0.323, implying the incorrect belief that $0.0234 = 0.234$. Previous studies have also reported that 9th graders and even pre-service teachers demonstrated this misconception in similar sorting tasks [20, 38]. Furthermore, students may still have this misconception even after abandoning others [13].

According to [24], there are four steps to redesign a tutor based on an improved cognitive model: (1) resequencing, (2) knowledge tracing, (3) creating new tasks, and (4) changing instructional messages, hint and feedback. Based on this framework and our analyses, we derived the following lessons for designing instructional materials in our digital learning game and other tutoring systems in decimal numbers:

1. Arrange the easy **Addition** problems (without or with one carry) at the beginning. The number of these easy problems can also be reduced, as over practice is already occurring based on the number of problems students are attempting with low error rates.
2. Design more **Addition** problems with varying difficulties (those with more carries are more difficult) and

position them in increasing order of difficulty.

3. Leave the operand fields blank in **Addition** problems so that students can practice aligning decimal digits. Getting feedback on this alignment task could in turn help them solve **Sequence** problems better.
4. Provide more scaffolding in **Sorting** problems, by first asking students to perform pairwise comparisons of the given numbers, then having them place the numbers in order. The first task can be used to track misconceptions and the second to track the skill of ordering.
5. Design questions in other problem types besides **Sorting** (e.g., **NumberLine**, **Bucket**) that address the *role of zero* misconception, as it may be stronger and persist longer than other misconceptions.

4.3 Advantages of Post-hoc KC Modeling

While, in general, KC modeling methods can be applied to any domain, domain knowledge is still critical for the interpretation of the improved models and an understanding of the newly discovered KCs. We have shown that we can apply methods in a post-hoc manner to a dataset in an educational domain to both achieve a better understanding and create a better fitting KC model. Our findings also demonstrate that the type of KC modeling we used can help guide changes to the types, contents and order of problems that are used in a decimal learning game (and educational technology more generally). From a theoretical perspective, the search space for a KC model in a given domain will be somewhere between a Single KC model, where every step represents the same KC, to a Unique Step model, where every step has its own KC. If we include the option of tagging a single step with multiple KCs, the space could get infinitely larger, but in a practical sense multi-coded steps could be combined to a single KC by concatenating the KCs on a given step. Several automated processes have been applied to create KC models by searching the possible space, such as Q-Matrix search [48], but they have the limitation of creating models with unlabeled skills. The methods that we used do not face this problem because we started with a fully labeled model and worked from there. Using visual and computational analyses on the learning curves, we were able to make improvements by combining the output of fitting models with domain knowledge. The original **Addition** KC is an excellent example of this approach in action. While the overall curve did show a declining error rate, every four opportunities looked as if the steps were getting harder (see Figure 3). Methodologically, this was a clear opportunity for improvement and likely a feature where each successive step in a problem became harder. Sure enough, this was the case as each of four problem steps required a carry, and the hardest problem required two carries. This is one example which demonstrates that we were able to not only get a better fitting model, but also attain a deeper domain understanding.

4.4 Future Work

In our next study, we will use the best KC model from this work as a test of how well it performs with a new population of students. There is also potential in connecting our work with earlier studies of student agency in digital learning games. In particular, [37] and [19] reported that even though students in the high-agency condition could choose to play any mini-game in any order, they did not learn more than those in the low-agency condition, who played a fixed

number of mini-games in a default order. [19] speculated that the former might be focused on selecting mini-games based on their visual themes (e.g., *Haunted House*, *Wild West* - see Figure 1) rather than learning content. To address this issue, we could employ an open learner model [4] that displays the estimated mastery level of each decimal skill to the students, where the skills are the KCs in our best model. In this scenario, we expect that students who exercise agency would be able to make informed selections of mini-games based on an awareness of their learning progress.

At the same time, digital learning games are intended to engage students and promote learning. Therefore, we want to explore the interactions between enjoyment and learning, particularly in how best to balance them. Just as learning can be modeled by knowledge components, can enjoyment also be modeled by “fun components,” and how would they be identified? We believe our digital learning game is an excellent platform for this exploration, because each mini-game has a separate learning factor (the decimal question) and enjoyment factor (the visual theme and game mechanics). It is also possible to track students’ enjoyment either through in-game surveys or automated affect detectors [1]. As our next step, we will design two study conditions, one that employs a traditional open learner model and one that captures and reflects students’ enjoyment, using the five problem types (worded in a more playful way, e.g., **Shooting** instead of **Sorting**, because all **Sorting** mini-games involve shooting objects such as spaceship) as the initial fun components. Findings from this follow-up study would then allow us to refine our enjoyment model and provide insights into whether a learning-driven or enjoyment-driven game design yields better outcomes.

In the direction of KC modeling, as mentioned in [19] and [52], it is possible that the the game contains more learning materials than required for mastery, or that some students may have exhibited greater learning efficiency than others. With the KC model identified in this work, we can then apply Bayesian Knowledge Tracing [11] to assess students’ mastery of each KC and verify the presence of learning efficiency or over-practice. Another area we plan to study is whether individual differences among the students in their gameplay and learning could lead to further improvement in predicting skill mastery based on the best-fit KC model, similar to previous research done in an intelligent tutor for genetics learning [15]. These individual differences could be accounted for by other features in the game outside of the identified cognitive-defined KCs [16].

5. CONCLUSION

Previous work has been done on refining KC models for educational systems in the manner we have shown here [51], although our research focused on the application of the refinement techniques to a digital learning game. We found that modeling KCs by problem types yields a better fit than modeling by the underlying misconceptions that were being tested. Furthermore, the refined KC model also showed us how to improve the original learning materials, in particular by focusing on the more challenging and persistent misconceptions, such as those involving multiple carries, role of zero and negative numbers. More generally, we demonstrated how learning curve analysis can be employed to perform

post-hoc KC modeling in a tutoring system with various types of task. In turn, our work opens up further opportunities to explore the interaction of student models with learning, enjoyment and agency, which would ultimately contribute to the design of a learning game that can adaptively balance these aspects.

6. ACKNOWLEDGEMENTS

This work was supported by NSF Award #DRL-1238619. The opinions expressed are those of the authors and do not represent the views of NSF. Special thanks to J. Elizabeth Richey and Erik Harpstead for offering valuable feedback. Thanks to Scott Herbst, Craig Ganoe, Darlan Santana Farias, Rick Henkel, Patrick B. McLaren, Grace Kihumba, Kim Lister, Kevin Dhou, John Choi, and Jimit Bhalani, for contributions to the development of the Decimal Point game.

7. REFERENCES

- [1] R. Baker, S. Gowda, M. Wixon, J. Kalka, A. Wagner, A. Salvi, V. Aleven, G. Kusbit, J. Ocumpaugh, and L. Rossi. Sensor-free automated detection of affect in a cognitive tutor for algebra. In *Educational Data Mining 2012*, 2012.
- [2] R. Baker, M. J. Habgood, S. E. Ainsworth, and A. T. Corbett. Modeling the acquisition of fluent skill in educational action games. In *International Conference on User Modeling*, pages 17–26. Springer, 2007.
- [3] J. Beck and X. Xiong. Limits to accuracy: How well can we do at student modeling? In *Educational Data Mining 2013*. Citeseer, 2013.
- [4] S. Bull and T. Nghiem. Helping learners to understand themselves with a learner model open to students, peers and instructors. In *Proceedings of workshop on individual and group modelling methods that help learners understand themselves, International Conference on Intelligent Tutoring Systems*, volume 2002, pages 5–13. Citeseer, 2002.
- [5] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis—a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pages 164–175. Springer, 2006.
- [6] H. Cen, K. R. Koedinger, and B. Junker. Is over practice necessary?—improving learning efficiency with the cognitive tutor through educational data mining. *Frontiers in artificial intelligence and applications*, 158:511, 2007.
- [7] M. T. Chi, N. De Leeuw, M.-H. Chiu, and C. LaVancher. Eliciting self-explanations improves understanding. *Cognitive science*, 18(3):439–477, 1994.
- [8] C. Conati, A. Gertner, and K. Vanlehn. Using bayesian networks to manage uncertainty in student modeling. *User modeling and user-adapted interaction*, 12(4):371–417, 2002.
- [9] C. Conati and M. Manske. Evaluating adaptive feedback in an educational computer game. In *International workshop on intelligent virtual agents*, pages 146–158. Springer, 2009.
- [10] C. Conati and X. Zhao. Building and evaluating an intelligent pedagogical agent to improve the effectiveness of an educational game. In *Proceedings of the 9th international conference on Intelligent user interfaces*, pages 6–13. ACM, 2004.
- [11] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [12] M. DeWolf, M. Bassok, and K. J. Holyoak. From rational numbers to algebra: Separable contributions of decimal magnitude and relational understanding of fractions. *Journal of experimental child psychology*, 133:72–84, 2015.
- [13] K. Durkin and B. Rittle-Johnson. The effectiveness of using incorrect examples to support learning about decimal magnitude. *Learning and Instruction*, 22(3):206–214, 2012.
- [14] K. Durkin and B. Rittle-Johnson. Diagnosing misconceptions: Revealing changing decimal fraction knowledge. *Learning and Instruction*, 37:21–29, 2015.
- [15] M. Eagle, A. Corbett, J. Stamper, B. M. McLaren, R. Baker, A. Wagner, B. MacLaren, and A. Mitchell. Exploring learner model differences between students. In *International Conference on Artificial Intelligence in Education*, pages 494–497. Springer, 2017.
- [16] M. Eagle, A. Corbett, J. Stamper, B. M. McLaren, A. Wagner, B. MacLaren, and A. Mitchell. Estimating individual differences for student modeling in intelligent tutors from reading and pretest data. In *International Conference on Intelligent Tutoring Systems*, pages 133–143. Springer, 2016.
- [17] J. P. González-Brenes and J. Mostow. Dynamic cognitive tracing: Towards unified discovery of student and cognitive models. *International Educational Data Mining Society*, 2012.
- [18] E. Harpstead and V. Aleven. Using empirical learning curve analysis to inform design in an educational game. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, pages 197–207. ACM, 2015.
- [19] E. Harpstead, J. E. Richey, H. Nguyen, and B. M. McLaren. Exploring the subtleties of agency and indirect control in digital learning games. In *International Learning Analytics & Knowledge Conference*. Springer, 2019.
- [20] J. Hiebert and D. Wearne. Procedures over concepts: The acquisition of decimal number knowledge. *Conceptual and procedural knowledge: The case of mathematics*, pages 199–223, 1986.
- [21] S. Isotani, D. Adams, R. E. Mayer, K. Durkin, B. Rittle-Johnson, and B. M. McLaren. Can erroneous examples help middle-school students learn decimals? In *European Conference on Technology Enhanced Learning*, pages 181–195. Springer, 2011.
- [22] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5):757–798, 2012.
- [23] K. R. Koedinger, E. A. McLaughlin, and J. Stamper. Automated student model improvement. *International Educational Data Mining Society*, 2012.
- [24] K. R. Koedinger, J. C. Stamper, E. A. McLaughlin, and T. Nixon. Using data-driven discovery of better student models to improve student learning. In *International Conference on Artificial Intelligence in*

- Education*, pages 421–430. Springer, 2013.
- [25] M. Y. Lai and S. Murray. What do error patterns tell us about hong kong chinese and australian students: understanding of decimal numbers? 2014.
 - [26] R. V. Lindsey, M. Khajah, and M. C. Mozer. Automatic discovery of cognitive skills to improve the prediction of student learning. In *Advances in neural information processing systems*, pages 1386–1394, 2014.
 - [27] R. Liu and K. R. Koedinger. Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains. *Journal of Educational Data Mining*, 9(1):25–41, 2017.
 - [28] R. Liu, E. A. McLaughlin, and K. R. Koedinger. Interpreting model discovery and testing generalization to a new dataset. In *Educational Data Mining 2014*. Citeseer, 2014.
 - [29] D. Lomas, D. Ching, E. Stampfer, M. Sandoval, and K. Koedinger. ” battleship numberline”: A digital game for improving estimation accuracy on fraction number lines. *Society for Research on Educational Effectiveness*, 2011.
 - [30] M. Manske and C. Conati. Modelling learning in an educational game. In *AIED*, pages 411–418, 2005.
 - [31] J. Martin and K. VanLehn. Student assessment using bayesian nets. *International Journal of Human-Computer Studies*, 42(6):575–591, 1995.
 - [32] R. E. Mayer. *Computer games for learning: An evidence-based approach*. MIT Press, 2014.
 - [33] B. M. McLaren, D. M. Adams, and R. E. Mayer. Delayed learning effects with erroneous examples: a study of learning decimals with a web-based tutor. *International Journal of Artificial Intelligence in Education*, 25(4):520–542, 2015.
 - [34] B. M. McLaren, D. M. Adams, R. E. Mayer, and J. Forlizzi. A computer-based game that promotes mathematics learning more than a conventional approach. *International Journal of Game-Based Learning (IJGBL)*, 7(1):36–56, 2017.
 - [35] K. Muldner and C. Conati. Evaluating a decision-theoretic approach to tailored example selection. In *IJCAI*, pages 483–488, 2007.
 - [36] A. Newell and P. S. Rosenbloom. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, 1(1981):1–55, 1981.
 - [37] H. Nguyen, E. Harpstead, Y. Wang, and B. M. McLaren. Student agency and game-based learning: A study comparing low and high agency. In *International Conference on Artificial Intelligence in Education*, pages 338–351. Springer, 2018.
 - [38] I. J. Putt. Preservice teachers ordering of decimal numbers: When more is smaller and less is larger!. *Focus on learning problems in mathematics*, 17(3):1–15, 1995.
 - [39] L. B. Resnick, P. Nesher, F. Leonard, M. Magone, S. Omanson, and I. Peled. Conceptual bases of arithmetic errors: The case of decimal fractions. *Journal for research in mathematics education*, pages 8–27, 1989.
 - [40] A. Roche and D. M. Clarke. When does successful comparison of decimals reflect conceptual understanding. *Mathematics education for the third millennium: Towards 2010*, pages 486–493, 2004.
 - [41] D. Rohrer, R. F. Dedrick, and K. Burgess. The benefit of interleaved mathematics practice is not limited to superficially similar kinds of problems. *Psychonomic bulletin & review*, 21(5):1323–1330, 2014.
 - [42] J. P. Rowe and J. C. Lester. Modeling user knowledge with dynamic bayesian networks in interactive narrative environments. In *Sixth AI and Interactive Digital Entertainment Conference*, 2010.
 - [43] R. J. Salden, V. A. Alevan, A. Renkl, and R. Schwonke. Worked examples and tutored problem solving: redundant or synergistic forms of support? *Topics in Cognitive Science*, 1(1):203–213, 2009.
 - [44] M. Schneider and E. Stern. The developmental relations between conceptual and procedural knowledge: A multimethod approach. *Developmental psychology*, 46(1):178, 2010.
 - [45] J. M. Schraagen, S. F. Chipman, and V. L. Shalin. *Cognitive task analysis*. Psychology Press, 2000.
 - [46] V. J. Shute, L. Wang, S. Greiff, W. Zhao, and G. Moore. Measuring problem solving skills via stealth assessment in an engaging video game. *Computers in Human Behavior*, 63:106–117, 2016.
 - [47] K. Stacey, S. Helme, and V. Steinle. Confusions between decimals, fractions and negative numbers: A consequence of the mirror as a conceptual metaphor in three different ways. In *PME CONFERENCE*, volume 4, pages 4–217, 2001.
 - [48] J. Stamper, T. Barnes, and M. Croy. Extracting student models for intelligent tutoring systems. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 1900. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.
 - [49] J. Stamper, K. Koedinger, R. Baker, A. Skogsholm, B. Leber, J. Rankin, and S. Demi. Plsc datashop: A data analysis service for the learning science community. In *International Conference on Intelligent Tutoring Systems*, pages 455–455. Springer, 2010.
 - [50] J. Stamper, K. Koedinger, and E. McLaughlin. A comparison of model selection metrics in datashop. In *Educational Data Mining 2013*, 2013.
 - [51] J. Stamper and K. R. Koedinger. Human-machine student model discovery and improvement using datashop. In *International Conference on AI in Education*, pages 353–360. Springer, 2011.
 - [52] Y. Wang, H. Nguyen, E. Harpstead, J. Stamper, and B. M. McLaren. How does order of gameplay impact learning and enjoyment in a digital learning game? In *International Conference on Artificial Intelligence in Education*. Springer, 2019.
 - [53] Y. Wang and R. S. Siegler. Representations of and translation between common fractions and decimal fractions. *Chinese Science Bulletin*, 58(36):4630–4640, 2013.
 - [54] M. Wilson and P. De Boeck. Descriptive and explanatory item response models. In *Explanatory item response models*, pages 43–74. Springer, 2004.

Predicting the Quality of Collaborative Problem Solving Through Linguistic Analysis of Discourse

Joseph M. Reilly
Harvard Graduate School of Education
13 Appian Way
Cambridge, MA 02138
1 (617) 496-5164
josephreilly@fas.harvard.edu

Bertrand Schneider
Harvard Graduate School of Education
13 Appian Way
Cambridge, MA 02138
1 (617) 496-2094
bertrand_schneider@gse.harvard.edu

ABSTRACT

Collaborative problem solving in computer-supported environments is of critical importance to the modern workforce. Coworkers or collaborators must be able to co-create and navigate a shared problem space using discourse and non-verbal cues. Analyzing this discourse can give insights into how consensus is reached and can estimate the depth of their understanding of the problem. This study uses Coh-Metrix, a natural language processing tool that measures cohesion, to analyze participant discourse from a recent multi-modal learning analytics study where novice programmers collaborated to use a block-based programming language to instruct a robot on how to solve a series of mazes. We significantly correlated thirty-five Coh-Metrix indices from the transcripts of dyads' discourse with collaboration, learning gains, and multimodal sensor values. We then fit a variety of machine learning classifiers to predict collaboration using the indices generated by Coh-Metrix as features. This study paves the way for real-time detection of (un)productive interactions from multimodal data and could lead to real-time interventions to support collaborative learning.

Keywords

Collaboration, computer-supported collaborative work, multi-modal learning analytics, Coh-Metrix.

1. INTRODUCTION

Collaborative problem solving with computer-based or computer-supported environments has long been a focus of research on educational technologies [1] and is now seen as a 21st century learning objective of critical importance to the workforce [2]. Discourse of collaborators who co-create and navigate a shared problem space can give insights into how consensus is reached and the depth of their understanding. Because qualitative coding of transcripts or video is laborious and time-consuming, capturing and quantifying the quality of social interactions remains a challenge in the social sciences.

Joseph Reilly and Bertrand Schneider "Predicting the Quality of Collaborative Problem Solving Through Linguistic Analysis of Discourse" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 149 - 157

Applied natural language processing can automate the analysis of large corpora of human language and is a foundational technique of educational data mining. Coh-Metrix [3], a tool originally developed to measure text difficulty, has been used to evaluate online discussion transcripts and intelligent tutoring system dialogue. By applying the tool to discourse from a collaborative problem-solving activity, we hypothesize that certain markers can indicate the quality of the collaboration and could be used to predict how well groups work together from their speech patterns.

This paper uses Coh-Metrix to analyze participant discourse from a recent multi-modal learning analytics (MMLA) [7] study where novice programmers used a block-based programming language to program a robot to solve a series of mazes [4]. Preliminary results from this study indicate the importance of speech equity and talking time; however, the full transcripts of the discourse have not yet been analyzed. In this paper, we explore multiple indices from the transcripts that are correlated with collaboration, learning gains, and multimodal sensor values. We then explore ways of predicting collaboration using the indices generated by Coh-Metrix as features fed into a variety of machine learning classifiers. Finally, we discuss these results and conclude with future avenues for this research.

2. LITERATURE REVIEW

A few decades ago, computer-supported collaborative learning (CSCL) emerged as a statement against the over-individualization of educational technology, emphasizing that collaborative learning can be fostered by carefully designed computer-supported activities [5]. Collaboration analysis in computer-supported environments has explored what facets of collaborative processes are essential for successful problem-solving and learning [29] with a particular emphasis on what aspects of these analyses can be automated [30]. When studying the process of collaboration, a 'Joint Problem Space' emerges that takes the form of a socially-negotiated knowledge structure that combines goal-setting, descriptions of the problem, and available actions [1]. This problem space can be understood through discourse analysis in conjunction with any other data collected on group behaviors.

In the search for new tools to analyze and model interactions between collaborators [6], Multi-Modal Learning Analytics (MMLA) has emerged. Sensors continue to get cheaper and easier to use while providing rich streams of data which can be used in conjunction to track and assess collaboration [7]. Data from multiple high frequency sensors can triangulate difficult to measure constructs and enhance overall predictive performance [31]. Analyzing features engineered from sensor data as well as dyadic

discourse provides a deeper view of the joint problem space, which includes nonverbal communication, posture, eye gaze, and arousal, among other possible metrics.

Coh-Metrix is an online tool that measures 106 linguistic features related to text easability, cohesion, lexical sophistication, syntactic complexity, and readability [3]. To differentiate between coherence and cohesion, the developers of this tool view cohesion as a quality of the text or discourse that can be directly measured, while coherence is in the mind of the reader [8]. The Coh-Metrix indices generally indicate the presence or absence of cohesive cues that tie the text together and make it easier or harder to understand. The tool focuses on local and overall text cohesion versus global cohesion [9].

Coh-Metrix has mainly been used in analyzing text readability and writing quality but has been applied successfully to other domains as well. Its indices have been used to detect lying in online discourse with one group member specified as sender and other as the receiver [10]. Tutor dialogue in AutoTutor was compared to naturalistic dialogue with a human tutor using Coh-Metrix to see how the dialogues differed on cohesion indices [11]. Indices for cohesion were also used to train affect detectors for AutoTutor users with the intention of developing real-time affect detectors based on cohesion [12]. The tool has also been used with online discussion transcripts to classify online discourse for levels of cognitive presence, and a classifier using Coh-Metrix features outperformed a similar algorithm using bag-of-words features [13].

Rarely do Coh-Metrix studies use transcripts of oral dialogue or assume participants are both novices (i.e. the task is not an expert-novice tutoring scenario.) Additionally, these indices have been sparingly used in MMLA research. In an MMLA study on a similar collaborative task, verbal coherence positively correlated with learning gains and significantly differed by condition [14]. Researchers then used language metrics to predict learning gains via support vector machine (SVM). Initial work from this current study has indicated that amount of talking and equity of talk time may be important indicators of good collaboration [15] but this discourse has not been analyzed in-depth yet.

3. RESEARCH QUESTIONS

This study attempts to answer the following research questions (RQs):

RQ1: Are Coh-Metrix indices derived from transcripts of discourse between co-located partners related to the quality of their

collaboration and learning gains?

RQ2: Are Coh-Metrix indices different across experimental conditions?

RQ3: Are Coh-Metrix indices associated with MMLA measures (e.g., Joint visual attention, physiological synchrony, nonverbal behaviors) that were previously significantly correlated with collaboration quality?

RQ4: What Coh-Metrix indices are most meaningful for estimating a group's collaboration?

RQ5: Can Coh-Metrix indices be used to train supervised machine learning algorithms to predict collaboration quality?

4. METHODS

4.1 The Study

Participants with no self-reported prior programming or robotics knowledge ("novices") were paired randomly with an unknown partner and tasked with programming a robot to solve a series of increasingly complex mazes in 30 minutes. During the activity, mobile eye-trackers recorded participant gaze data, bracelets captured electrodermal activity, and a motion sensor collected movement and position data. A 2x2 study design was employed to test two different collaboration interventions: an informational intervention that described the benefits of collaborating on tasks and a visualization intervention that graphically plotted relative verbal contributions from each participant from the previous 30 seconds. The informational intervention is the primary on discussed here as the other did not result in significant differences in dependent measures. All participants gained knowledge of basic programming skills according to a pre-post survey ($t = 6.18$, $p < 0.001$) and a 7 percentage point increase in collaboration quality was associated with a 2 percentage point increase in code quality when controlling for gender and prior education ($p < 0.001$). For more details of experimental design and overall results, see Table 1 and [4]. Figure 1 shows a typical image of the experiment in progress.

4.2 Participants

Forty-two dyads completed the study and the first sessions each researcher conducted were removed to improve overall fidelity ($N = 40$ groups). Participants were drawn from an existing study pool at a university in New England in the United States. 62% of participants reported being students at the university of various levels, with ages ranging from 19 to 51 years old with a mean age

Table 1. Summary of measures from study.

Independent Measures	Process Data	Dependent Measures
Control Condition: no intervention	Eye-tracking: Joint visual attention by dyads on different areas of interest, amounts of time looking at areas of interest	Expert ratings of collaboration
Treatment Condition: informational intervention orally delivered by researcher prior to beginning of main portion of study.	Electrodermal activity: differences between individuals, synchrony measures, rates of change	Task performance measures
	Movement and posture: proximity, alignment, bimanual coordination, total movement, leaning, synchrony measures	Survey gains



Figure 1. Experimental setup from the study.

of 27 years. 60% of participants identified as female. Participants were paid \$20 per 90-minute session of the study.

4.3 Procedure

Prior to the main activity, participants signed informed consent paperwork and took a 5-minute pre-survey pertaining to simple programming tasks. Once the survey was complete, sensors were applied to the participants and calibrated while the function of each sensor was explained.

Next, participants were shown a tutorial video explaining how to write code in Tinker, a block-based environment designed for use with the sensors and motors on the robot. Participants were then given 5 minutes to write a simple program to move the robot forward past a red line on the table in front of them. After the completion of this tutorial activity, participants were shown a second tutorial video that explained more advanced features of the programming environment like setting threshold sensor values for triggering commands and using conditional statements. A reference sheet was also provided to participants that summarized the content of both tutorial videos.

At this point, groups in the Intervention condition were read a summary of several research findings relevant to collaboration such as the importance of equity of speech time in high quality collaboration. Dyads in the Control group were given no such information after completing the tutorial ($N = 20$ groups in each condition).

Dyads then had 30 minutes to write code to guide their robot through a series of increasingly complex mazes. Participants did not know the arrangement of the mazes ahead of time and were prompted to write code that could solve any simple maze. Once the robot completed a maze twice successfully, the next one was provided by the researcher. During this portion of the activity, predetermined hints were provided to groups at 5-minute intervals to ensure common pitfalls identified in pilot testing were avoided. Following the completion of this portion, the post-survey was administered, demographic data was collected, sensors were removed, and participants were paid and debriefed.

4.4 Dependent Measures

Dyads' collaboration was evaluated live by the researcher conducting the session. Quality of collaboration was assessed on nine different scales derived from Meier, Spada, and Rummel's work on assessing collaboration in CSCL [16]: sustaining mutual understanding, dialogue management, information pooling, reaching consensus, task division, time management, technical coordination, reciprocal interaction, and individual task orientation.

Each scale was on a -2 to 2 scale, and all scales were added together to generate an overall collaboration rating for dyads. Multiple researchers conducted sessions of the study and thus coded dyads' behavior. Researchers double coded 20% of the sessions from videos collected during the session and achieved an inter-rater reliability of $\alpha = 0.65$ (75% agreement).

Learning of computational skills (identifying a bug in block-based code, anticipating the output of a code segment, describing how to do a task with pseudocode, etc.) was assessed individually via pre- and post-tests with four questions each. These measures were adapted from [17, 18]. Researchers coded a subset of the responses to 100% agreement based on their demonstrations of understanding of computational thinking principles then coded the remaining surveys with the developed rubric.

4.5 Data Pre-Processing

Data collected by the eye-trackers, wristbands, and motion sensors each needed to be processed individually prior to merging for analysis. See Reilly, Ravenell, and Schneider for details on the processing of the Kinect motion sensor data [15] and Dich, Reilly, and Schneider for use of the electrodermal activity data from the Empatica E4 bracelet [19]. Four different physiological synchrony measures were calculated for movement and electrodermal (EDA) data: Signal Matching (SM), Instantaneous Derivative Matching (IDM), Directional Agreement (DA) and Pearson's Correlation (PC). SM was calculated as the differences in area between the plots of the team members' EDA, IDM calculates how closely the slopes of the physiological signal curves match, DA identifies whether individuals' signal data increase or decrease at the same time, and PC looks for a linear relationship between EDA data of both participants. For details on the calculation of these measures, see [19].

For use in this study, the eye-tracking data was summarized in two different ways: The proportion of time both participants were looking at the same spot (joint visual attention, see [20]) and the proportion of time spent looking at various areas of interest around the room (the computer screen, the maze, the robot, etc.) as determined by the fiducial markers placed on all objects in the experiment.

Audio recordings of sessions were transcribed using multiple iterations of assignments to Amazon Mechanical Turk workers until it was suitable for analysis. Transcripts were formatted via Python to match the requirements of Coh-Metrix and were sent to the Institute for Intelligent Systems at the University of Memphis for analysis. All analyses in this study were done in Python using the scikit-learn package [25].

5. RESULTS

5.1 RQ1: Correlations with Collaboration

Indices positively correlated with our rating of collaboration include more dialogue between partners (as measured by the number of sentences and words uttered), the use of adverbs, the CELEX word frequency (how often content words appear in sentences), and the familiarity of content words used. Deep cohesion (the use of causal connectives to signify causal relationships) and increased temporality (cues about temporality and tense) were also significantly positively correlated with collaboration. Additionally, the Coh-Metrix L2 readability score (use of simple grammar that an English language learner could

more easily parse) was also positively correlated. An appendix with definitions of all indices discussed here is provided at the end of this paper (also available in Appendix A of [3]). Significant correlations are listed in Table 2.

Table 2. Indices correlated with collaboration (red indicates negative correlation).

Index	Value (p-value)	Description
DESSC	0.51 (0.0087)	Descriptive indices that describe the number and length of paragraphs, sentences, and words
DESWC	0.735 (<0.0001)	
DESPL	0.51 (0.0087)	
DESWLsy	-0.27 (0.023)	
DESWLsyd	-0.36 (0.019)	
DESWLlt	-0.41 (0.0081)	
DESWLtd	-0.17 (0.016)	
PCNARz	0.53 (0.0042)	Text easability principal component scores
PCNARp	0.55 (0.0023)	
PCDCz	0.476 (0.039)	
PCTEMPz	0.53 (0.0068)	
PCTEMPp	0.52 (0.0064)	
PCCNCz	-0.41 (0.028)	
PCCNCp	-0.48 (0.0063)	
LDTTRc	-0.76 (<0.0001)	Lexical diversity (unique words per total number of words)
LDTTRa	-0.76 (>0.0001)	
LDVOD	-0.25 (0.047)	
CNCCaus	0.45 (0.037)	Incidence of connectives
CNCLogic	0.49 (0.042)	
CNCADC	0.36 (0.036)	
SMINTER	0.40 (0.044)	Ratio of intentional particles to intentional verbs
SYNNP	-0.43 (0.0012)	Number of modifiers per noun phrase, mean
DRPVAL	-0.33 (0.020)	Agentless passive voice density, incidence
WRDNOUN	-0.43 (0.0016)	Word information (part of speech category, syntactic categories)
WRDADV	0.47 (0.028)	
WRDPRP2	-0.68 (0.0005)	
WRDFRQc	0.43 (0.0004)	
WRDFRQa	0.16 (0.038)	
WRDAOAc	-0.28 (0.048)	
WRDFAMc	0.52 (0.0008)	
WRDCNCc	-0.46 (0.0063)	
WRDIMGc	-0.51 (0.007)	
WRDHYPv	-0.49 (0.0047)	
WRDHYPnv	-0.44 (0.0020)	
RDL2	0.36 (0.023)	Coh-Metrix L2 Readability

Indices negatively correlated with collaboration include the mean number of syllables per word, the number of nouns, the lexical diversity (unique number of total words), and hypernymy for nouns and verbs (using specific words instead of general ones.) Features that indicate the difficulty of understanding text are generally negatively correlated with collaboration, such as modifiers per noun phrase (complex syntax places higher demands on working memory), agentless passive voice, and the ratio of intentional

particles to intentional verbs (a higher ratio indicates more inference is needed to understand the text.) Indices for specificity of content words are also negatively correlated, such as concreteness and imageability (how easy it is to create a mental image of the word.)

5.2 RQ1: Correlations with Learning Gains

The magnitude of participant learning gains on the pre-post survey was also correlated significantly with several Coh-Metrix indices. Unlike collaboration, learning gain is positively correlated with lexical diversity ($r = 0.41$, $p = 0.041$) which indicates that use of specific language may aid learning of computer science principles. On the other hand, learning gain is negatively correlated with pronoun incidence ($r = -0.44$, $p = 0.041$) and referential cohesion ($r = -0.34$, $p = 0.049$) indicating that use of vague, overlapping language by the dyad is associated with lower gains on the survey.

5.3 RQ3: Differences by Condition

Differences between the Control and Intervention conditions could also be seen in their discourse during the activity. According to paired t-tests, groups in the Control condition had fewer words ($t = -3.4$, $p = 0.0019$), fewer adverbs ($t = -2.30$, $p = 0.029$), fewer sentences ($t = -2.17$, $p = 0.038$) and shorter paragraphs ($t = -2.17$, $p = 0.038$) than those in the Intervention condition. Additionally, the Control condition discourse had higher lexical diversity than the Intervention condition (LDTTRc: $t = 2.93$, $p = 0.0065$; LDTTRa: $t = 3.03$, $p = 0.0049$) which was shown above to be negatively correlated with collaboration.

With respect to the differences between groups that saw a visualization intervention that plotted relative verbal contributions from each participant, only the L2 Readability score differed significantly between conditions ($t = -2.16$, $p = 0.039$). As this intervention did not result in significant differences in our outcome measures, it makes sense that the impact on our Coh-Metrix indices is also minimal.

5.4 RQ4: Correlations with MMLA Values

We also explored whether any dyad-level features engineered from our sensor data might correlate with the Coh-Metrix indices that were previously seen to be significantly related to collaboration. Out of 30 features from our Kinect, eye-tracking, and EDA data, four were significantly correlated with three or more indices shown in Table 2. Correlation coefficients for these four features are shown in Figure 2.

From our eye-tracking data, the amount of time participants spent looking together at neither the maze nor the computer (aoi_0) was significantly negatively correlated with word length ($r = -0.42$, $p = 0.034$), lexical diversity ($r = -0.40$, $p = 0.042$), and syntactic complexity ($r = -0.39$, $p = 0.049$). Those three indices were also negatively correlated with collaboration. The amount of time participants spent looking at the maze and robot but not the computer (aoi_5) was positively associated with second person pronouns ($r = 0.65$, $p < 0.001$) and L2 readability ($r = 0.39$, $p = 0.047$) but negatively related to word length ($r = -0.42$, $p = 0.034$), word diversity ($r = -0.47$, $p = 0.016$), and use of the passive voice ($r = -0.40$, $p = 0.046$). This also follows a similar pattern to our correlations with collaboration.

The directional agreement (DA) of the dyad is calculated as the proportion of time where EDA for both participants was increasing or decreasing at the same time (in other words, it is a measure of

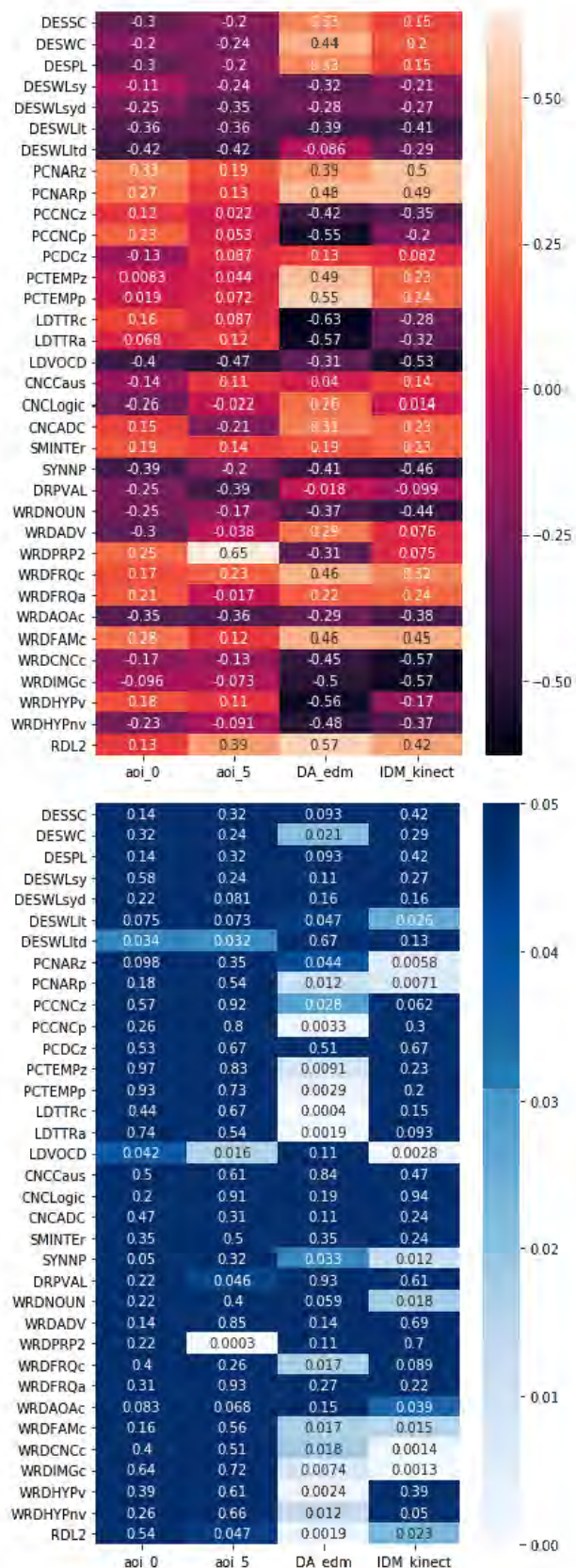


Figure 2. Correlation coefficients (top) and p-values (bottom) for select MMLA features.

physiological synchrony). Higher DA was positively associated with the number of words used ($r = 0.44$, $p = 0.021$), narrativity ($r = 0.48$, $p = 0.012$), temporality ($r = 0.55$, $p = 0.003$), word frequency ($r = 0.46$, $p = 0.017$), word familiarity ($r = 0.45$, $p = 0.017$), and L2 readability ($r = 0.57$, $p = 0.002$). DA was significantly negatively associated with lexical diversity ($r = 0.44$, $p = 0.021$), syntactic complexity ($r = -0.41$, $p = 0.033$), word concreteness ($r = -0.45$, $p = 0.018$), imageability of content words ($r = -0.50$, $p = 0.007$), and hypernymy ($r = -0.56$, $p = 0.002$). The directions of these correlations also fit with what we observed.

The Instantaneous Derivative Matching (IDM) of the Kinect movement data is calculated as the proportion of time where movement of both dyad members is either increasing or decreasing at a similar rate. IDM of movement was positively associated with narrativity ($r = 0.50$, $p = 0.006$), word familiarity ($r = 0.45$, $p = 0.015$), and L2 readability ($r = 0.42$, $p = 0.023$). Movement IDM was negatively correlated with word length ($r = -0.41$, $p = 0.026$), lexical diversity ($r = -0.53$, $p = 0.003$), syntactic complexity ($r = -0.46$, $p = 0.012$), the number of nouns used ($r = -0.44$, $p = 0.018$), word concreteness ($r = -0.57$, $p = 0.001$), imageability of content words ($r = -0.57$, $p = 0.001$), and hypernymy ($r = -0.37$, $p = 0.049$). Again, these correlations are of similar magnitude and direction as those seen in our results from collaboration.

5.5 RQ5: Predicting Collaboration

In order to explore how we might be able to use the Coh-Metrix indices to predict quality of collaboration, we classified dyads in terms of their collaboration ratings using a variety of typical machine learning classifiers. All 106 Coh-Metrix indices were used as features to classify the 40 groups. Missing values were imputed with their column means and all features were normalized prior to their use.

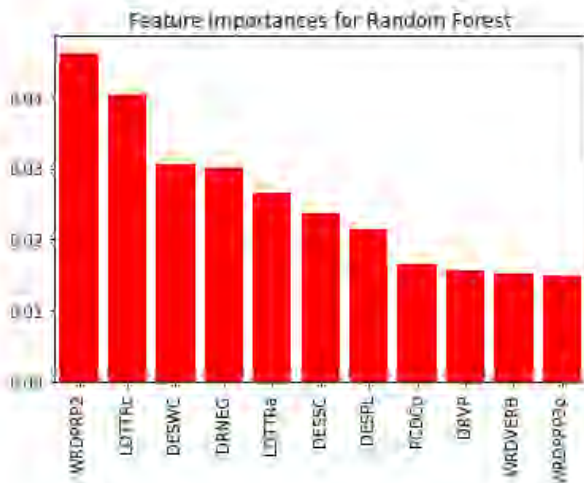
We first separated our participants into two groups based on the median value of group collaboration. We trained a Naïve Bayes classifier, a support vector machine (SVM), and a Random Forest (RF) model [21] on our entire data. NB usually works well with text data [26], SVM excels at binary classification [27], and RF along with other tree-based classifiers have been used successfully in the EDM community with a wide variety of educational datasets [28]. These algorithms were also selected as they are computationally rapid to implement once tuned and may be used in a real-time nature during a future intervention. The alpha, loss, and penalty used by SVM as well as the number of estimators, maximum depth, and criterion function for RF were selected by grid search with 5-fold cross-validation (CV). To address issues of overfitting with a small sample size, we report both training accuracy as well as the highest average accuracy achieved by our 5-fold cross-validation.

As shown in Table 2, the Random Forest model outperformed both Naïve Bayes and SVM on the binary classification median split task. While the 100% train accuracy of the RF is surely due to overfitting, the high CV accuracy. All algorithms were able to outperform random assignment by large margins. We next separated out participants into four groups based on the quartile values of group collaboration. Our three models were fit with the same procedure and once again RF outperformed both other algorithms on the training data. When looking at validation results, however, the RF and SVM classifiers performed identically. Simpler models may avoid the overfitting issues leading to the reported 100% training accuracy.

Table 2. Accuracy of classifiers.

	Median Split Train	Median Split CV	Quartile Split Train	Quartile Split CV
Naïve Bayes	0.88	0.74	0.81	0.51
SVM	0.88	0.75	0.84	0.53
Random Forest	1.00	0.84	1.00	0.53

To gain more insight into how these classifiers made their assignments, we investigated which features the RF model for the quartile split problem was ranking as most important for making assignments. Figure 3 plots the eleven most important features for our classification problem. Beyond that point, the feature importance rankings are too similar to derive insight. It is important to note that importance here is agnostic of whether these features correspond to good or bad collaboration; they are simply the most meaningful for deciding between them. Second person pronoun incidence (WRDPRP2) and lexical diversity (LDTTRc, as measured by the type-token ratio [22]) are the most important features. Word count (DESWC), sentence count (DESSC), incidence of negation expressions (DRNEG), text easability (PCDCp), and verb phrase density (DRVP) also rate highly. Many of these features were previously seen to be significantly correlated with collaboration, but verb phrase density and the incidence of negations appear here in our analyses for the first time.

**Figure 3. Feature importances for the quartile split Random Forest model.**

6. DISCUSSION

In general, our findings indicate that our strongest collaborating dyads communicated more in terms of amount of words and sentences as well as the length of each utterance before the other participant would interject. In addition, these groups used more abstraction when referring to content words and terms and employed basic words and grammar to convey meaning in a direct fashion. They avoided using the passive voice or pronouns while reaching a consensus on a simple shared set of words to describe the task and their actions. While synonyms and extraneous

modifiers were not used by strong collaborators, adverbs were used to define particular actions the robot needed to perform, and the use of logical, causal, and temporal connectives indicates a value to explicitly linking actions across space and time to meet the desired outcome. These indices of cohesion jointly allow collaborators to negotiate a shared problem space regardless of English language proficiency or level of education.

To ground the above findings, here is an example of a low collaborating dyad's discourse regarding programming the robot for a new maze:

A: So let's, we can do, no, yeah, we can put up, if yeah and if it's that then it goes this.

B: Then we add, turn right.

A: Yeah it will go right and then it will take for, wait for 10 seconds and then take a left. Also take a left.

B: Yeah, go, go forward. Go forward. Left, then we go straight.

A: Let's go forward. Left and then right. Then left and right.

In contrast, this is dialogue from a high collaborating group at a similar point in the activity:

C: So let's try changing this value to...greater than the second "If Do".

D: Okay. I just want to see if, oh, what did I do there, I just want to see if that what difference that makes.

C: Perfect. All right, are you ready?

D: Yep. Nope, all right.

C: Okay so, we've got it going forward and turning right so at least the right works. That one's correct now.

D: Now... if we change this number so let's go back to the widget.

C: Okay, okay. I think I've got it, so we needed to turn so when we got it turn right, we need to maybe check to if it turns left or right needs to be "greater than".

In the second dyad's discourse, more complete yet simple grammar and explicit markers of turn-taking result in a much fuller discourse that is easier to track. Their use of causal language and conditionals and implies a greater grasp of the content of the activity.

The importance and effects of cohesion are typically much higher for low-knowledge readers, with this relationship dubbed the "reverse cohesion effect" in discourse literature [23]. As none of our participants had any prior knowledge of robotics or computer programming, the importance of cohesion in participant discourse is likely to be crucial in similar educational settings. Reading skill and young age can also interact with this effect but these issues were not confounders in our study due to the use of oral dialogue and our population being solely adults.

As far as how the Treatment and Control groups differed, the Control groups typically communicated less (fewer sentences with shorter exchanges), used less adverbs, and had a higher lexical diversity (which was shown to be negatively correlated with collaboration). This difference shows that even simple verbal cues delivered as an intervention prior to an activity can have a positive association with collaboration by fostering more cohesive communication. The effect size of an informational intervention such as this on collaboration might be effectively used as a baseline when comparing more elaborate interventions in similar activities in the future. It is also reassuring to see the low effect on the Coh-Metrix indices from the visualization intervention condition that had no effect on collaboration. This validates our strategy of using

these indices without coding scheme to assess collaboration quality in a variety of different experimental conditions.

Comparing average learning gains on the pre-post survey to the Coh-Metrix indices is difficult for several reasons. First, the survey was done at the individual level and by only using the mean change we ignore when participants unequally learned during the task. Second, gains may be susceptible to ceiling effects where high gains are not seen due to high performance on the pre-test. Third, the dyads were instructed to program the robot to solve mazes and thus their conversation revolved around that task. The activity certainly utilized the computer science principles that were assessed in the survey, but the discussion was not as specific as a tutor dialogue regarding programming.

Despite these issues and challenges, several Coh-Metrix indices reveal what types of markers in the discourse can signal learning taking place. While lexical diversity was negatively associated with collaboration, it appears to be beneficial for learning. Knowing and applying more terms for phenomena or problem-solving strategies may aid participants transfer their knowledge from the experimental task to the post-survey. Additionally, too much referential cohesion may make ideas difficult to separate out of context and thus more challenging to use in isolation on test questions.

It is worth noting that features engineered from all three of our MMLA sensors provided insight into how to assess collaboration using the Coh-Metrix indices identified as significantly related to collaboration. When joint visual attention fell outside of the tabletop or laptop (aoi_0), this could generally be interpreted as participants looking at each other (as relatively little time was spent with both participants simultaneously looking at the facilitator, the same spot on the wall, or anything unrelated to the task). The proportion of time spent doing this negatively correlated with word length, lexical diversity, and syntactic complexity (all of which are markers of poor collaboration). Eye contact is positively associated with problem solving and facilitates conceptual understanding in group settings [24] so this result triangulates established literature findings. Joint visual attention being more focused on the maze and robot instead of the laptop (aoi_5) correlated positively with second person pronoun use and readability while negatively correlating with word length, word diversity, and use of the passive voice. This can be interpreted as dyads communicating with each other more effectively by looking at the physical problem space and talking through the steps needed to solve the problem versus spending more time in the programming interface editing code.

Two of our measures of synchrony (directional agreement for EDA and instantaneous derivative matching for motion) are positively associated with indices deemed good for collaboration and negatively correlated with indices seen to be negatively related to collaboration. By focusing on these four features from our multimodal data, we might be able to automatically assess collaboration during trials, which could be used to provide formative feedback and design new interventions based on these measurements.

Finally, we used supervised machine learning algorithms in hopes of being able to detect and intervene while dyads are working together. The relative feature importances from our Random Forest classifier also shed light on what indices are most useful for assessing collaboration. Second person pronoun incidence, number

of words, and lexical diversity appeared in our correlations with collaboration, while verb phrase density and the incidence of negations did not appear in our previous results. The emergence of negation in this model will need to be studied more thoroughly. Increased incidence of negation expressions may signal discord between the participants that could hinder the joint construction of meaning en route to problem solving. It is possible that the relationships between these features and collaboration are nonlinear and are thus not detected as readily by simple correlational analyses. Additionally, the overfitting of the models may be due to the lack of regularization of model complexity. With the range of -2 to 2 for the collaborative scoring, it might be more appropriate to fit a regression model instead of classifying the scores.

This preliminary work has several limitations that must temper the results. The small sample size of 40 groups leads to overfitting of our classifiers and may interfere with the ability of some of the Coh-Metrix algorithms to function. The designers recommend using a corpus of roughly 300 texts of 300 words each to study text easibility [3]. While the length of our transcripts exceeds these recommendations, it is unclear what effect our sample may have on this novel use of Coh-Metrix. We collected no reading level demographic data on our participants, nor did we ask for whom was English a first language. Additionally, this preliminary work does not address the important issue of how does communication (and thus collaboration) differ when participants have very different expressive language capabilities.

The developers of Coh-Metrix intended these indices to be the “low-hanging fruit” of computational linguistics, choosing to use simple metrics instead of complex computational linguistic models [3]. While this will likely be valuable for developing real-time dynamic interventions that can’t be slowed down by computationally expensive operations, we need to compare these results to more complex models and other natural language processing methods. Future work will also explore “driver-passenger” models that investigate emergent leadership behavior and uneven talk time in the discourse as well as the role of eye contact in the quality of collaboration.

7. CONCLUSION

This research paves the way for real-time detection of (un)productive interactions from multimodal data, potentially facilitating the development of fail-soft real-time interventions to support collaborative learning. While Coh-Metrix is only available currently as an online service, similar analytical platforms can be run locally [9] and could offer advice based on specific issues detected in the discourse rather than general distribution of talk time. These indices of cohesion and easibility have proven to be versatile and serve as effective features for estimating the rough quality of dyadic discourse with regard to collaboration quality.

8. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1748093. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

- [1] Roschelle, J. and Teasley, S.D., 1995. The construction of shared knowledge in collaborative problem solving.

- In *Computer Supported Collaborative Learning* (pp. 69-97). Springer, Berlin, Heidelberg.
- [2] National Research Council. 2012. *Education for Life and Work: Developing Transferable Knowledge and Skills in the 21st Century*. Washington, DC: The National Academies Press.
 - [3] McNamara, D.S., Graesser, A.C., McCarthy, P.M. and Cai, Z., 2014. *Automated Evaluation of Text and Discourse with Coh-Metrix*. Cambridge University Press.
 - [4] Starr, E., Reilly, J., and Schneider, B. 2018. Using Multi-Modal Learning Analytics to Support and Measure Collaboration in Co-Located Dyads. In *Proceedings of the 13th International Conference on the Learning Sciences*, 448–455.
 - [5] Dillenbourg, P., Järvelä, S. and Fischer, F., 2009. The evolution of research on computer-supported collaborative learning. In *Technology-Enhanced Learning*, 3-19. Springer, Dordrecht.
 - [6] Dillenbourg, P., Baker, M.J., Blaye, A. and O'Malley, C., 1995. The evolution of research on collaborative learning. Spada, E. and Reiman, P. *Learning in Humans and Machine: Towards an interdisciplinary learning science.*, Elsevier, Oxford, 189-211.
 - [7] Blikstein, P. and Worsley, M., 2016. Multimodal Learning Analytics and Education Data Mining: using computational technologies to measure complex learning tasks. *Journal of Learning Analytics*, 3(2), 220-238.
 - [8] Graesser, A.C., McNamara, D.S., Louwerse, M.M. and Cai, Z., 2004. Coh-Metrix: Analysis of text on cohesion and language. *Behavior research methods, instruments, & computers*, 36(2), 193-202.
 - [9] Crossley, S.A., Kyle, K. and McNamara, D.S., 2016. The tool for the automatic analysis of text cohesion (TAACO): Automatic assessment of local, global, and text cohesion. *Behavior research methods*, 48(4), 1227-1237.
 - [10] Duran, N.D., Hall, C., McCarthy, P.M. and McNamara, D.S., 2010. The linguistic correlates of conversational deception: Comparing natural language processing technologies. *Applied Psycholinguistics*, 31(3), 439-462.
 - [11] Graesser, A.C., Jeon, M., Yan, Y. and Cai, Z., 2007. Discourse cohesion in text and tutorial dialogue. *Information Design Journal*, 15(3), 199-213.
 - [12] D'Mello, S.K., Dowell, N. and Graesser, A.C., 2009, July. Cohesion Relationships in Tutorial Dialogue as Predictors of Affective States. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, 9-16.
 - [13] Kovanović, V., Joksimović, S., Waters, Z., Gašević, D., Kitto, K., Hatala, M. and Siemens, G., 2016, April. Towards automated content analysis of discussion transcripts: A cognitive presence case. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*, 15-24.
 - [14] Schneider, B. and Pea, R. 2015. Does seeing one another's gaze affect group dialogue? A computational approach. *Journal of Learning Analytics*, 2(2), 107-133.
 - [15] Reilly, J., Ravenell, M., and Schneider, B. 2018. Assessing Collaboration Using Motion Sensors and Multi-Modal Learning Analytics. In K.E. Boyer & M. Yudelso (Eds.), *Proceedings of the 11th International Conference on Educational Data Mining*, 24 –257.
 - [16] Meier, A., Spada, H., and Rummel, N. 2007. A rating scheme for assessing the quality of computer-supported collaboration processes. *Computer Supported Learning*, 2, 63–86.
 - [17] Brennan, K. and Resnick, M. 2012. New frameworks for studying and assessing the development of computational thinking. Presented at the Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.
 - [18] Weintrop, D. and Wilensky, U. 2015. Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. Presented at the 11th Annual ACM Conference on International Computing Education Research.
 - [19] Dich, Y., Reilly, J., and Schneider, B. 2018. Using Physiological Synchrony as an Indicator of Collaboration Quality, Task Performance and Learning. In *Proceedings of the 19th International Conference on Artificial Intelligence in Education*, 98 – 110.
 - [20] Richardson, D.C. and Dale, R., 2005. Looking to understand: The coupling between speakers' and listeners' eye movements and its relationship to discourse comprehension. *Cognitive Science*, 29(6), 1045-1060.
 - [21] Breiman, L., 2001. Random forests. *Machine Learning*, 45(1), 5-32.
 - [22] Templin, M. 1957. Certain language skills in children: Their development and interrelationships. The University of Minnesota Press, Minneapolis.
 - [23] O'Reilly, T. and McNamara, D.S., 2007. Reversing the reverse cohesion effect: Good texts can be better for strategic, high-knowledge readers. *Discourse processes*, 43(2), 121-152.
 - [24] Joiner, R., Scanlon, E., O'Shea, T., Smith, R.B. and Blake, C., 2002, January. Evidence from a series of experiments on video-mediated Collaboration: Does Eye Contact Matter?. In *Proceedings of the Conference on Computer Support for Collaborative Learning: Foundations for a CSCL Community*, 371-378.
 - [25] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12(Oct), 2825-2830.
 - [26] Rennie, J.D., Shih, L., Teevan, J. and Karger, D.R. 2003. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the 20th international conference on machine learning*, 616-623.
 - [27] Scholkopf, B. and Smola, A.J. 2001. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

- [28] Romero, C. and Ventura, S., 2007. Educational data mining: A survey from 1995 to 2005. *Expert systems with applications*, 33(1), 135-146.
- [29] Rummel, N. and Spada, H. 2005. Learning to collaborate: An instructional approach to promoting collaborative problem solving in computer-mediated settings. *The Journal of the Learning Sciences*, 14(2), 201-241.
- [30] Dillenbourg, P., Järvelä, S., and Fischer, F. 2009. The evolution of research on computer-supported collaborative learning. In *Technology-enhanced learning*, 3-19. Springer, Dordrecht.
- [31] Ochoa, X. and Worsley, M. 2016. Augmenting Learning Analytics with Multimodal Sensory Data. *Journal of Learning Analytics*, 3(2), 213-219.

10. Appendix: Coh-Metrix Indices

Index	Description	Index	Description
DESSC	Sentence count, number of sentences	CNCLogic	Logical connectives incidence
DESWC	Word count, number of words	CNCADC	Adversative and contrastive connectives incidence
DESPL	Paragraph length, number of sentences, mean	SMINTER	Ratio of intentional particles to intentional verbs
DESWLsy	Word length, number of syllables, mean	SYNNP	Number of modifiers per noun phrase, mean
DESWLsyd	Word length, number of syllables, standard deviation	DRVP	Verb phrase density, incidence
DESWLlt	Word length, number of letters, mean	DRNEG	Negation density, incidence
DESWLltd	Word length, number of letters, standard deviation	DRPVAL	Agentless passive voice density, incidence
PCNARz	Text Easability PC Narrativity, z score	WRDNOUN	Noun incidence
PCNARp	Text Easability PC Narrativity, percentile	WRDADV	Adverb incidence
PCDCz	Text Easability PC Deep cohesion, z score	WRDPRP2	Second person pronoun incidence
PCTEMPz	Text Easability PC Temporality, z score	WRDFRQc	CELEX word frequency for content words, mean
PCTEMPp	Text Easability PC Temporality, percentile	WRDFRQa	CELEX Log frequency for all words, mean
PCCNCz	Text Easability PC Word concreteness, z score	WRDAOAc	Age of acquisition for content words, mean
PCCNCp	Text Easability PC Word concreteness, percentile	WRDFAMc	Familiarity for content words, mean
LDTTRc	Lexical diversity, type-token ratio, content word lemmas	WRDCNCc	Concreteness for content words, mean
LDTTRa	Lexical diversity, type-token ratio, all words	WRDHYPv	Hypernymy for verbs, mean
LDVOD	Lexical diversity, VOD, all words	WRDHYPv	Hypernymy for nouns and verbs, mean
CNCCaus	Causal connectives incidence	RDL2	Coh-Metrix L2 Readability

Grade Prediction Based on Cumulative Knowledge and Co-taken Courses

Zhiyun Ren
Computer Science
George Mason University
4400 University Drive,
Fairfax, VA 22030
zren4@gmu.edu

Xia Ning
Biomedical Informatics
The Ohio State University
Columbus, OH 43210
Xia.Ning@osumc.edu

Andrew S. Lan
College of Information and
Computer Sciences
University of Massachusetts
Amherst
140 Governors Dr., Amherst,
MA 01003
andrewlan@cs.umass.edu

Huzefa Rangwala
Computer Science
George Mason University
4400 University Drive,
Fairfax, VA 22030
rangwala@cs.gmu.edu

ABSTRACT

Over the past decade, low graduation and retention rates have plagued higher education institutions. To help students graduate on time and achieve optimal learning outcomes, many institutions provide advising services supported by educational technologies. Accurate grade prediction is an integral part of these services such as degree planning software, personalized advising systems and early warning systems that can identify students at-risk of dropping from their field of study. In this work, we present next-term grade prediction models based on students' cumulative knowledge and co-taken courses. The proposed models are based on a matrix factorization framework and incorporate a co-taken course interaction function to learn the influence from the co-taken courses on the target course. The co-taken course interaction function is formed by a neural network, which takes the knowledge difference between the co-taken courses and the target course as input, and outputs an influence value that will be used to predict students' grades on the target course. The experimental results on various datasets from a U.S. University demonstrate that the proposed models significantly outperform competitive baselines across different test sets. Furthermore, we analyze the proposed models' performance with different numbers of co-taken courses as well as different numbers of co-taken course subjects, and highlight with an application case study how a student might make decisions related to selection of courses. The codes are available at <https://github.com/Zhiyun0411/EDM>.

Keywords

matrix factorization, next-term grade prediction, cumulative

Zhiyun Ren, Xia Ning, Andrew Lan and Huzefa Rangwala "Grade Prediction Based on Cumulative Knowledge and Co-taken Courses" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 158 - 167

knowledge, co-taken courses

1. INTRODUCTION

For over a decade higher education institutions in the United States have been grappling with low graduation rates [9]. The National Center for Education Statistics ¹ reports that approximately 59% of students who started college in 2009 were able to graduate and obtain a 4-year college program degree within 6 years. There is a pressing need for data-driven applications and services to guide students through academic pathways and achieve better learning outcomes. Many higher education institutions have implemented programs and services supported by educational technologies to increase overall graduation rates [17]. For example, Academic Advising service ² provides effective student-centered advising at Purdue University. Graduation Progression Success (GPS) Advising ³ implemented at Georgia State University helps identify at-risk students and have advisors respond alerts. Their reports show a 6% increase of 6-year graduation rate over 4 years. Our work aims to help students select courses for the next term by developing methods that can provide accurate grade prediction for the courses they have not taken yet.

In the past few years, many approaches have been developed for next-term grade prediction. One of the most popular approaches is matrix factorization (MF), which is inspired from the Recommender Systems (RS) literature [2, 3, 7, 15, 18]. Specifically, MF decomposes the student-course grade matrix into two matrices containing student and course latent factors, respectively. The predicted grade of a student on a course is given by the inner product of the corresponding student and course latent factors [4, 10]. There are other extended MF-based models which achieve better grade prediction results than MF. For example, Morsy *et al.* [8] proposed a Cumulative Knowledge-based Regression Model (CK) to tackle the next-term grade prediction problem. CK models each student with cumulative knowledge acquired by the student in the

¹<https://nces.ed.gov>

²<http://www.purdue.edu/advisors/index.html>

³<http://giving.gsu.edu/student-success/>

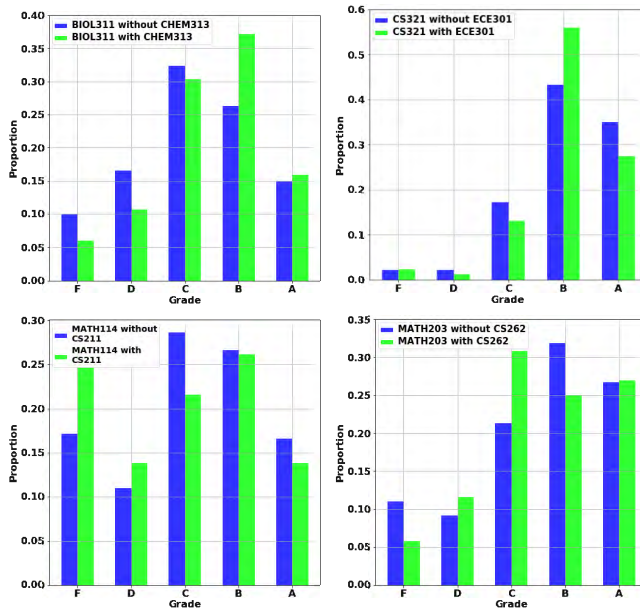


Figure 1: Students’ Performance with Different Co-taken Course Pairs. Note: BIOL311 is course “General Genetics”. CHEM313 is course “Organic Chemistry”. CS321 is course “Software Engineering”. ECE301 is course “Digital Electronics”. MATH114 is course “Analytic Geometry and Calculus”. CS211 is course “Object Oriented Programming”. MATH203 is course “Linear Algebra”. CS262 is course “Low-level Programming”.

past terms. However, among all the existing methods for next-term grade prediction [2, 13, 14], very few consider the effect of co-taken courses on students’ performance.

We conduct a statistical analysis on a dataset collected from George Mason University in order to demonstrate the effects of co-taken courses on students’ performance. Figure 1 shows the true grade distribution of students’ on a specific course *with* and *without* enrolling in another course in the same term. The course pairs we choose in this analysis are frequently co-occurring in our dataset. For each target course pair, we choose the students who take more than four courses in a term, including the corresponding course pairs. We keep the students if the other co-taken courses only share few topics/material as the target course pairs. Figure 1 shows that students who take BIOL311 (Genetics) with CHEM313 (Organic Chemistry) have fewer “F”, “D” and “C” grades, and several more “B” grades than those students who only take BIOL311 in a term. Similar trend has been found for course pairs CS321 (Software Engineering) and ECE301 (Digital Electronics). Moreover, students who take MATH114 (Calculus) with CS211 (Object Oriented Programming) will have more “F” grades than those students who only take MATH114 in a term. Students who co-take MATH203 (Linear Algebra) and CS262 (Low-level programming) have more “C” grades than those students who only take MATH203 in a term. This shows that it can be challenging for students to take some courses together in a term (e.g., MATH114 and CS211, MATH203 and CS262), while it might not cause grade drop if taking other course pairs together (e.g., BIOL311 and CHEM313, CS321 and ECE301). Thus, we assume that co-taken courses can have substantial effect on student grades in different ways.

In this work, we propose grade prediction models that incorporate

both Cumulative Knowledge and Co-taken Courses (CKCC) to predict students’ performance in the next term. Inspired by Morsy *et al.* [8], the proposed methods model each student’s latent factors by cumulating the knowledge provided by the sequence of courses the student has taken in the past terms. Furthermore, we introduce a co-taken course interaction function to model the influence of the co-taken courses on students’ performance. The co-taken course interaction function is formed by a neural network which takes the knowledge difference between the co-taken courses and the target course as input, and outputs an influence value from the co-taken courses on the target course. We conduct comprehensive experiments on various datasets collected from George Mason University and thorough analysis on the effect of co-taken courses. Our experimental results show that CKCC significantly outperforms other competitive baselines methods for the task of grade prediction. We also provide detailed case study on how our model can help student in course selection for the next term.

The main contributions can be summarized as follows:

1. We develop CKCC models on next-term grade prediction. The models consider both students’ cumulative knowledge and co-taken courses in the target term. To the best of our knowledge, this is the first work that learns and explicitly incorporates influences from co-taken courses for grade prediction.
2. We provide a detailed case study on how our model helps students in course selection for the next term by comparing the performance of CKCC with different sets of co-taken courses.

2. RELATED WORK

2.1 Grade Prediction Approaches

Methods originating from recommender systems research have attracted increasing attention in educational data mining [2, 3, 13, 14, 20]. Sweeney *et al.* [18, 19] applied several recommender systems approaches to predict next-term grades. The authors implemented MF-based methods including SVD, SVD-kNN and factorization machine and simple baseline methods including global, student, and course means. The work showed that MF-based methods consistently achieve better grade prediction results over the baselines. Elbadrawy *et al.* [1] developed a domain-aware grade prediction method with student/course-group biases. This method groups students based on majors and academic levels. Additionally, it groups courses based on course levels and course subjects. The method assumes that the students/courses in a same group tend to have similar biases. Accordingly, this method models biases for each student and course group within a MF framework and achieved significant improvement on grade prediction performance over baselines.

2.2 Grade Prediction based on Student Historical Information

Polyzou *et al.* [12] addressed the future course grade prediction problem with different approaches based on sparse linear models and MF approaches. The experimental results showed that the course-specific regression approach achieved the best performance among all approaches. This method predict a student’s performance using a sparse linear combination of the grades that the student obtained in past courses. Morsy *et al.* [8] proposed a model named Cumulative Knowledge-based Regression Model (CK) to predict student’s grade on a certain course at the next term. CK

models each student with the cumulative knowledge he/she obtained from the sequence of courses he/she took in the past. Then CK calculated the inner product of the cumulative knowledge vector of a student and the required knowledge vector of the target course as the predicted grade. The experimental results showed that CK significantly outperforms MF in grade prediction. Ren *et al.* [13] proposed a matrix factorization model with temporal course-wise influence to predict next term student grades. This model considers two components in predicting a student's grade on a certain course: (i) the student's competence with respect to the target course's topics, content and requirements, etc., and (ii) student's previous performance over other courses. The study concluded that considering temporal influence can significantly improve the next-term grade prediction performance.

2.3 Neural Network in Educational Data Mining

Neural networks have been applied to solve many educational data mining problems. For example, Sharma *et al.* [16] proposed a composite deep neural network to predict whether the educational video is lively or not. The proposed method first used a convolutional neural network to extract the video features, and then used a deep recurrent neural network to predict the human movement label in order to detect video liveliness. Klingler *et al.* [6] presented a semi-supervised classification pipeline that employed deep variational auto-encoders to detect students who are suffering from developmental dyscalculia. Piech *et al.* [11] introduced Deep Knowledge Tracing (DKT) to model student learning with Recurrent Neural Networks. The authors provided experiments on how to use DKT to detect latent structure between the assessments in the dataset. The models proposed in this paper tackle the challenges of next-term grade prediction with students' history information (the sequence of courses the student has taken) and the co-taken courses in the next term. The main contribution of our model is to explicitly incorporate the co-taken courses with in MF framework.

3. PRELIMINARIES AND PROBLEM DEFINITION

3.1 Problem Definition

Formally, student-course grades will be represented by G_1, G_2, \dots, G_T for a total of T terms. Each G_t is a matrix, and contains the set of student-course grades for all students enrolled in courses within term t . For all the students, the set of student-course grades up to term t can be represented by $G^t = \bigcup_{i=1}^t G_i$. The set of courses that student s has taken in term t is represented by $C_{s,t}$ and the set of grades that student s achieves in term t is represented by $G_{s,t}$. The set of courses that student s has taken up to term t is represented by C_s^t , and the set of grades that student s has achieved up to term t is represented by G_s^t .

In this paper, all vectors are represented by bold lower-case letters and all matrices are represented by upper-case letters. Row vectors are represented by having the transpose superscript^T, otherwise by default they are column vectors. A predicted value is denoted by having a $\tilde{\cdot}$ symbol. Table 1 summarizes the key notations used in this paper.

Given student-course grades up to term $t - 1$ and the set of courses each student plans to take at term t , the objective of our work is to predict student's grades on a specific course given the set of co-taken courses at term t .

3.2 Grade Prediction based on Matrix Factorization

MF methods factor the student-course grade matrix into two matrices containing latent factors of courses and students in a common knowledge space, respectively [1, 12]. The dimension of the knowledge space is much lower than that of the original student-course grade matrix. We use \mathbf{p}_s ($\mathbf{p}_s \in \mathbb{R}^k$) and \mathbf{q}_c ($\mathbf{q}_c \in \mathbb{R}^k$) to represent latent factors of k dimensions for student s and course c , respectively. Thus, the grade of student s on course c can be predicted as

$$\tilde{g}_{s,c} = \mathbf{p}_s^T \mathbf{q}_c + b_s + b_c. \quad (1)$$

where b_s and b_c are bias terms for student s and course c , respectively.

3.3 Grade Prediction with Cumulative Knowledge

Morsy *et al.* [8] proposed the CK model which learns each student's latent factors with cumulative knowledge acquired by the student in past terms. Specifically, CK uses two vectors to model a course: the provided knowledge by the course and the prerequisite knowledge of the course, respectively. A student's latent factor is given by the knowledge accumulated from the previous course that the student has taken and the corresponding course grades. Formally, the cumulative knowledge acquired by student s up to term t is represented by $\mathbf{p}_{ck(s)}^t$, and is given by:

$$\mathbf{p}_{ck(s)}^t = \sum_{g_{s,c'} \in G_{s,t-1}} (e^{-\lambda(t-t_{s,c'})} \mathbf{k}_{c'} \cdot g_{s,c'}), \quad (2)$$

where $t_{s,c'}$ is the term in which student s took course c' , $e^{-\lambda(t-t_{s,c'})}$ is an exponential time decay function with $\lambda > 0$ denoting the decay rate, $\mathbf{k}_{c'}$ is the latent knowledge factor of course c' , and $g_{s,c'}$ is the grade of student s on course c' . Given $\mathbf{p}_{ck(s)}^t$, CK predicts student s 's grade on course c in term t as follows:

$$\tilde{g}_{s,c}^t = \mathbf{p}_{ck(s)}^t{}^T \mathbf{q}_c. \quad (3)$$

Note that in prior work, Ren *et al.* [14] have shown that CK can achieve better grade prediction performance when the cumulative knowledge $\mathbf{p}_{ck(s)}^t$ is averaged in Eq 3. Therefore, $\tilde{g}_{s,c}^t$ is presented as follows:

$$\tilde{g}_{s,c}^t = \frac{1}{|G_s^{t-1}|} \sum_{g_{s,c'} \in G_{s,t-1}} (e^{-\lambda(t-t_{s,c'})} \mathbf{k}_{c'} \cdot g_{s,c'})^T \mathbf{q}_c, \quad (4)$$

We refer to this model as the averaged cumulative knowledge (CK) model and will consider it as one of our baseline methods.

4. METHODS

4.1 Model Overview

In this paper, we propose grade prediction models that incorporate Cumulative Knowledge and Co-taken Courses (CKCC). To predict student s 's grade on course c in term t , CKCC takes into account two factors: i) cumulative knowledge of student s up to term $t - 1$, and ii) the other courses that will be taken together with course c in term t . To model the first factor, we adopt the CK model as in Eq. 4, that is, we cumulate the provided knowledge of the courses which student s has taken in the past, denoted as c' , to represent his/her cumulative knowledge, and use a latent factor to represent knowledge required by course c . To model the second factor, we introduce an co-taken course interaction function $f(\cdot)$ to learn the

Table 1: Notations

Notation	Explanation
m	number of courses
n	number of students
k	number of latent dimensions
$\mathbf{p}_{ck(s)}^t$	the cumulative knowledge of student s up to term t
\mathbf{q}_c	latent factor of the required knowledge components of course c
\mathbf{k}_c	latent factors of the provided knowledge components of course c
b_s	student bias term
b_c	course bias term
$g_{s,c}^t$	the grade of student s on course c at term t
$t_{s,c}$	the academic term when student s takes course c
G_t	student-course grades at term t
G^t	all the student-course grades up to term t
$G_{s,t}$	all the grades student s obtains at term t
G_s^t	all the grades student s obtains up to term t
$C_{s,t}$	the set of courses student s chooses at term t
C_s^t	the set of courses student s chooses up to term t

influence from co-taken courses, denoted as c'' , on student s 's grade on course c in term t .

Specifically, we use a latent vector \mathbf{q}_c to represent the knowledge components that course c requires. We hypothesize that the difference of the required knowledge between two courses will cause the influence from one course on the other, as shown in Figure 1. Based on this hypothesis, the difference between \mathbf{q}_c of course c and $\mathbf{q}_{c''}$ of a co-taken course c'' can be used in $f(\cdot)$ to learn the influence from c'' to c . We sum up the differences between each co-taken course c'' and c in order to aggregate the influence. Thus, the sum of the absolute values of the differences between each $\mathbf{q}_{c''}$ and \mathbf{q}_c , that is, $\sum_{c'' \in C_{s,t} \setminus \{c\}} |\mathbf{q}_{c''} - \mathbf{q}_c|$, is used in $f(\cdot)$ to learn the influence from all co-taken courses. Note that the use of absolute values here is to avoid the scenarios in which the influences from different co-taken courses are canceled out. Thus, CKCC predicts student s 's grade on course c in term t as follows:

$$\tilde{g}_{s,c}^t = \frac{1}{|G_{s,c}^{t-1}|} \sum_{g_{s,c'} \in G_{s,c}^{t-1}} (e^{-\lambda(t-t_{s,c'})} \mathbf{k}_{c'} \cdot g_{s,c'})^\top \mathbf{q}_c + f\left(\sum_{c'' \in C_{s,t} \setminus \{c\}} (|\mathbf{q}_{c''} - \mathbf{q}_c|)\right), \quad (5)$$

where $|\mathbf{q}_{c''} - \mathbf{q}_c|$ is the vector of absolute values of entry-wise difference between latent vector $\mathbf{q}_{c''}$ and latent vector \mathbf{q}_c , $c'' \in C_{s,t} \setminus \{c\}$ indicates that course c'' is one of courses taken together with c in term t . Note that in Eq. 5, the two terms share a common latent vector \mathbf{q}_c .

4.2 Co-taken Course Interaction Function

In CKCC, the co-taken course interaction function $f(\cdot)$ learns the influence on student s 's grade on course c from all the other co-taken courses in term t . We hypothesize that such influence can be nonlinear in general. Therefore, we use a feedforward neural network (FNN) [21] as $f(\cdot)$ to model the influence. The FNN takes the input as described in last section, and outputs a scalar influence value on course c . We use hyperbolic tangent (Tanh) as the activation function in each layer of the FNN. Note that when there are no hidden layers and no nonlinearity, the FNN model learns the weights directly from the input layer (i.e., difference of courses) to

Algorithm 1 CKCC: Learn

```

1: procedure CKCC_LEARN
2:   Initialize  $\mathbf{k}_c, \mathbf{q}_c$  for each  $c$ 
3:    $\eta \leftarrow$  learning rate
4:    $T \leftarrow$  number of terms in training set
5:    $\lambda \leftarrow$  time decay parameter
6:    $\alpha_1, \alpha_2, \alpha_3 \leftarrow$  regularization weight
7:    $t \leftarrow 2$ 
8:    $iter \leftarrow 0$ 
9:   while iter < maxIter do
10:    for  $t \leq T$  do
11:      for all  $g_{s,c}^t \in G_s^t$  do ▷ step 1
12:         $\hat{g}_{s,c}^t \leftarrow g_{s,c}^t - f(\sum_{c' \in C_{s,t} \setminus \{c\}} (|\mathbf{q}_{c'} - \mathbf{q}_c|))$ 
13:         $\mathbf{p}_{ck(s)} \leftarrow 0$ 
14:        for all  $c' \in C_s^{t-1}$  do
15:           $\mathbf{p}_{ck(s)} \leftarrow \mathbf{p}_{ck(s)} + e^{-\lambda(t_{s,c} - t_{s,c'})} \mathbf{k}_{c'} \cdot g_{s,c'}^{t_{s,c'}}$ 
16:           $\tilde{g}_{s,c}^t \leftarrow \mathbf{p}_{ck(s)}^\top \mathbf{q}_c$ 
17:           $e_{s,c}^t = \hat{g}_{s,c}^t - \tilde{g}_{s,c}^t$ 
18:          for all  $c' \in C_s^{t-1}$  do
19:             $\mathbf{k}_{c'} \leftarrow \mathbf{k}_{c'} + \eta(\mathbf{q}_c \cdot e^{-\lambda(t_{s,c} - t_{s,c'})} \cdot g_{s,c'} \cdot e_{s,c}^t - \alpha_1 \cdot \mathbf{k}_{c'})$ 
20:             $\mathbf{q}_c \leftarrow \mathbf{q}_c + \eta(\mathbf{p}_{ck(s)} \cdot e_{s,c}^t - \alpha_2 \cdot \mathbf{q}_c)$ 
21:          for all  $g_{s,c}^t \in G_s^t$  do ▷ step 2
22:             $\hat{g}_{s,c}^t \leftarrow g_{s,c}^t - \mathbf{p}_{ck(s)}^\top \mathbf{q}_c$ 
23:             $\tilde{g}_{s,c}^t \leftarrow f(\sum_{c' \in C_{s,t} \setminus \{c\}} (|\mathbf{q}_{c'} - \mathbf{q}_c|))$ 
24:             $e_{s,c}^t = \hat{g}_{s,c}^t - \tilde{g}_{s,c}^t$ 
25:            Update  $\Theta_f$  with Adam
26:           $iter \leftarrow iter + 1$ 
return  $\Theta = \{\{\mathbf{k}_c\}, \{\mathbf{q}_c\}\}, \Theta_f$ 

```

the output layer (i.e., the influence), and the function $f(\cdot)$ becomes a simple inner product operation (parameterized by a vector). This simplified model is referred to as CKCC- I . Figure 2 shows the structure of the CKCC model.

4.3 Optimization of CKCC

Given the grade estimation as in Equation 5, we formulate the grade prediction problem for term T as the following optimization problem:

$$\begin{aligned} \underset{\Theta, \Theta_f}{\text{minimize}} \quad & \sum_s \sum_{t=1}^{T-1} \sum_{g_{s,c}^t \in G_s^t} (g_{s,c}^t - \tilde{g}_{s,c}^t)^2 \\ & + \alpha_1 (|\mathbf{k}_c| + |\mathbf{q}_c|) + \alpha_2 (\|\mathbf{k}_c\|_2^2 + \|\mathbf{q}_c\|_2^2) \\ & + \alpha_3 \|\text{vec}(\Theta_f)\|_2^2, \end{aligned} \quad (6)$$

where $\Theta = \{\{\mathbf{k}_c\}, \{\mathbf{q}_c\}\}$ represents the set of latent vectors, and Θ_f represents the parameters of $f(\cdot)$. α_1 , α_2 , and α_3 denote the nonnegative weights on the regularization terms to prevent overfitting.

The optimization process for CKCC is presented in Algorithm 1. It consists of two steps: The first step is to update the course parameters, i.e., Θ , using stochastic gradient descent. The second step is to update $f(\cdot)$ parameters, i.e., Θ_f , with the adaptive moment estimation (Adam) algorithm [5].

5. EXPERIMENTS

5.1 Dataset Description

The data used in this work is obtained from George Mason University. Our dataset contains two student groups: first-time freshmen (FTF; i.e., students who begin their study initially at this Uni-

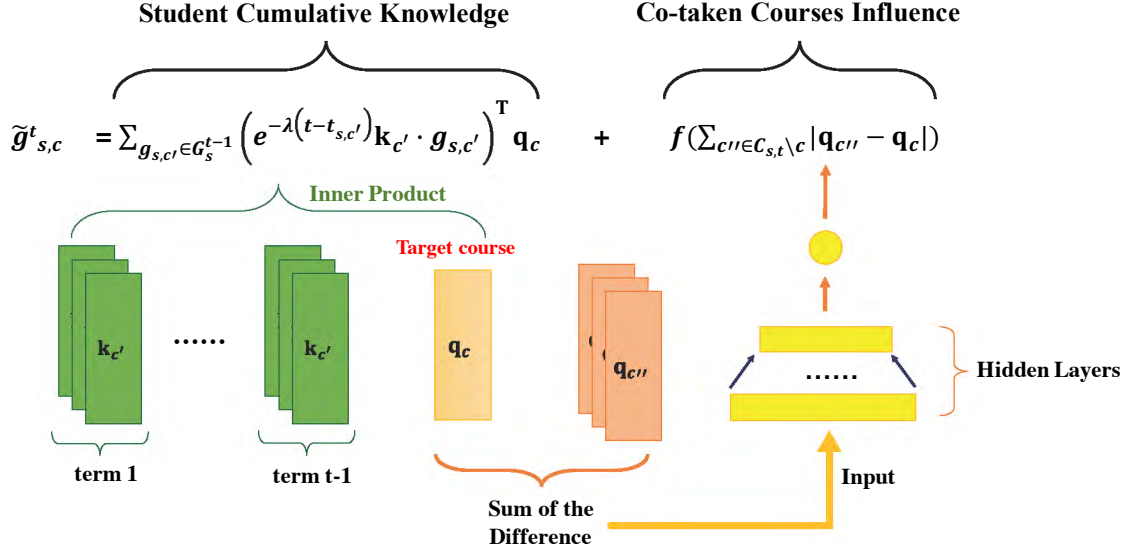


Figure 2: CKCC Model Structure

Table 2: Dataset Statistics

Major	FTF student group			TR student group		
	#S	#C	#S-C	#S	#C	#S-C
MATH	271	693	3,325	243	597	2,031
PHYS	144	488	2,044	73	286	905
CHEM	427	673	4,942	257	473	1,937
IT	430	473	5,984	1,163	487	10,302
CS	819	714	16,955	526	435	7,840
BIOL	1,951	1,197	22,065	1,481	980	10,851

#S, #C and #S-C are the number of students, courses and student-course pairs from Fall 2009 to Spring 2018, respectively.

versity), and transfer students (TR; i.e., students who transfer to this University from a different one). The dataset was extracted in the period of Fall 2009 to Spring 2018. It includes information of 23,435 FTF students and 28,470 TR students across 153 majors. For simplicity, we use students from six different majors to evaluate the proposed models. These majors have different numbers of enrolled students, courses, and different major syllabi. We will evaluate these majors on both FTF and TR student groups. The majors in our experiment include: (i) Mathematical Sciences (MATH), (ii) Physics (PHYS), (iii) Chemistry (CHEM), (iv) Information Technology (IT), (v) Computer Science (CS) and (vi) Biology (BIOL). Table 2 shows the statistics across these majors.

5.2 Experimental Protocols

To assess the performance of our next-term grade prediction models, we trained our models on data up to term $T-1$ and make predictions for term T . We evaluate our method for three test terms, i.e., Spring 2018, Fall 2017 and Spring 2017. As an example, for evaluating predictions for term Fall 2017, data from Fall 2009 to Spring 2017 is considered as training data and data from Fall 2017 is testing data. datasets. Figure 3 shows the three different train-test splits.

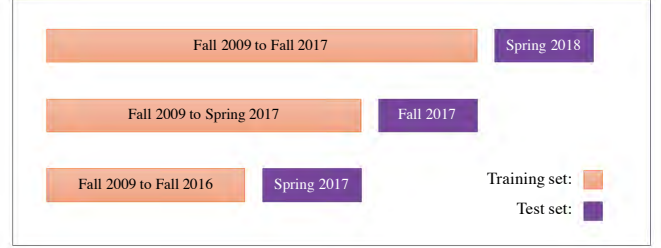


Figure 3: Different Experimental Protocols

5.3 Evaluation Metrics

In our experiments, we use Mean Absolute Error (MAE) to evaluate the predicted results in numbers. MAE is calculated as:

$$MAE = \frac{\sum_{g_{s,c}^t \in G_T} |g_{s,c}^t - \tilde{g}_{s,c}^t|}{|G_T|} \quad (7)$$

where $g_{s,c}^t$ and $\tilde{g}_{s,c}^t$ are the ground-truth grade and predicted grade for student s on course c at term T , respectively. G_T is the set of student-course grades in the T -th term, which is considered as the test set in our experiment.

Moreover, since a student receives a letter grade for a course, i.e., A, A-, ..., F, we use the Percentage of Tick Accuracy (PTA) [12] as one of our evaluation metrics. During training, we map letter grades "A+" and "A" to the real-valued grade point number 4.0, "A-" to 3.67, "B+" to 3.33, etc. During testing, we map the predicted grade point numbers back to their closest letter grades. Then, we define tick as the difference between two consecutive letter grades (e.g., C+ vs C or C vs C-). We then compute the percentage of predicted grades that match the actual grades (or within 0-ticks of them), and those that are within 1 tick and within 2 ticks of the actual grades as PTA_0 , PTA_1 , and PTA_2 , respectively.

5.4 Compared Methods

Since there is no prior research on the influence of co-taken courses within a same term, we use the two following methods and three other variants of CKCC as baselines in our experiments:

- **MF** The MF model is described as Eq. 1.
- **CK** The CK model is described as Eq. 4.
- **MFCC** We add the co-taken course influence to the MF model, and obtain the Matrix Factorization with Co-taken Courses (MFCC) model. Specifically, the predicted grade of student s on course c at term t is defined as

$$\hat{g}_{s,c}^t = \mathbf{p}_s^T \mathbf{q}_c + f\left(\sum_{c'' \in C_s^t \setminus c} (|\mathbf{q}_{c''} - \mathbf{q}_c|)\right), \quad (8)$$

where \mathbf{p}_s denotes the latent factors of for student s . Similar to the CKCC model, we optimize the MFCC model with two steps by alternately updating the latent factors and the model parameters in the mapping function $f(\cdot)$.

- **MFCC- l** The MFCC- l model is a special case of the MFCC model where $f(\cdot)$ is simply an inner product (parameterized by a vector) instead of an FNN.
- **CKCC- l** The CKCC- l model is described in Section 4.2.

5.5 Parameter Learning

The set of parameters in the optimization problem (Eq 6) includes the number of latent dimensions (i.e., k), regularization parameters (i.e., α_1 , α_2 , and α_3) and the decay rate (i.e., λ). We performed a grid search over all the parameters with $k \in \{5, 10, \dots, 25\}$, and $\alpha_1, \alpha_2, \alpha_3, \lambda \in \{1e-3, 1e-2, 0.1\}$. Note that for the CKCC and MFCC models, the optimal neural network structure (e.g., number of layers, the size of each layer) depends on the value of k . Thus, we swept different neural network structure parameters for every k value in our grid search. The neural network structures that consistently achieve good performance contain one hidden layer with 2 or 3 hidden units.

6. RESULTS AND DISCUSSION

6.1 Overall Performance

Table 3 and 4 shows the overall performance for all methods for both FTF and TR student groups, respectively.

Table 3 shows that for FTF students, CKCC and CKCC- l outperform the baseline methods over most datasets. Specifically, CKCC outperforms the other compared methods across different experimental protocols by 4.39%, 7.01%, 3.50%, 3.87% in terms of MAE, PTA₀, PTA₁, and PTA₂, respectively. Furthermore, CK based methods outperform MF based methods on all experimental protocols. This table also shows that co-taken course based methods (MFCC, MFCC- l and CKCC, CKCC- l) outperform their baseline methods (MF and CK) on all experimental protocols, respectively. This illustrates that for FTF students, both cumulative knowledge and co-taken courses have great influence on student's performance, and the proposed methods can capture such influence accurately.

Table 4 shows that CK has competitive results over TR students. Moreover, for MF based methods, MFCC and MFCC- l outperform MF for all the experimental protocols. This illustrates that co-taken courses are likely to have influence on student's performance, but

the influence may not be as strong as it is of cumulative knowledge for TR students.

6.2 Analysis on Individual Majors

In order to understand the proposed methods' performance on each major, we have tested all the aforementioned methods on different majors separately. We conducted this group of experiments for both FTF and TR students. And we use Spring 2018 as test set. We provide detailed experimental results in Table 5 and 6.

Table 5 shows that the CKCC model outperforms other compared methods for some majors (e.g., PHYS, CS) on all metrics, but has weak performance on some metrics for other majors (e.g., MATH, CHEM). Especially for MATH major, CKCC has the highest MAE result while MFCC and MFCC- l have the best MAE result. The reason might be that the performance of CKCC relies on the student historical information, and it tends to have good performance on the students with rich historical information. However, in the test set, some students in certain majors do not have much historical information and thus drag down the model performance. Table 6 shows that, for TR students, there is no method that consistently outperforms others across different metrics. The reason might be that the diversity in student characteristics (many TR students have different backgrounds) leads to diverse course selection plans among them. Such diversity greatly influences the performance of the different models.

6.3 Linear versus Nonlinear Mapping Function

As aforementioned, we have two forms of co-taken course interaction function: FNN model and linear model (parameterized by a vector). Specifically, we compare the results for MFCC versus MFCC- l , and CKCC versus CKCC- l , respectively, in order to understand how different mapping functions $f(\cdot)$ influence grade prediction performance. Table 3 shows that for FTF students, MFCC- l has slightly better performance than MFCC, and CKCC- l has competitive performance as CKCC across different experimental protocols. Same trend has shown in table 4 for TR students. Furthermore, table 5 shows that MFCC and CKCC consistently outperform MFCC- l and CKCC- l across different majors for FTF students. This illustrates that the influence of co-taken courses for FTF student group can be better captured by a nonlinear model (i.e., FNN) than a simple linear model. Table 6 shows that for TR students, MFCC and CKCC don't always outperform MFCC- l and CKCC- l for different majors. The reason might be that some TR students will have fewer co-taken courses than those of FTF students, and the influence from co-taken courses can be well captured by a linear model.

6.4 Performance on Different Numbers of Co-taken Courses

In this section, we test the CKCC model on different data subgroups with different number of co-taken courses in a term. Specifically, we take the students in the test set and divide them into five groups: students who take $\{2, 3, 4, 5, 6+\}$ courses (6+ refers to six and more). We perform this experiment on each major for both FTF and TR students, respectively. For the sake of page limit, we only show the results for FTF students. Figure 4 shows the experimental results in terms of PTA₀, PTA₁ and PTA₂. The results show that different majors exhibit different trends when the number of co-taken courses varies. For example, for CHEM and BIOL majors, the performance of the CKCC model on PTA improves with more

Table 3: Performance Comparison for All Methods on FTF students

Method	Spring 2018				Fall 2017				Spring 2017			
	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂
MF	0.762	0.172	0.303	0.549	0.759	0.168	0.303	0.556	0.772	0.162	0.306	0.540
MFCC- <i>l</i>	0.756	0.180	0.320	0.565	0.745	0.186	0.331	0.574	0.757	0.181	0.331	0.564
MFCC	0.763	0.175	0.317	0.573	0.753	0.188	0.322	0.573	0.760	0.173	0.317	0.565
CK	0.726	0.190	0.330	0.575	0.724	0.184	0.336	0.575	0.727	0.186	0.333	0.575
CKCC- <i>l</i>	0.711	0.189	0.338	0.589	0.712	0.191	0.343	0.589	0.717	0.182	0.332	0.587
CKCC	0.716	0.187	0.332	0.593	0.709	0.195	0.334	0.588	0.710	0.196	0.339	0.594

Table 4: Performance Comparison for All Methods on TR students

Method	Spring 2018				Fall 2017				Spring 2017			
	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂
MF	0.775	0.184	0.316	0.537	0.760	0.157	0.300	0.565	0.773	0.168	0.299	0.550
MFCC- <i>l</i>	0.763	0.178	0.315	0.543	0.748	0.187	0.326	0.571	0.755	0.185	0.328	0.563
MFCC	0.761	0.174	0.321	0.544	0.754	0.177	0.330	0.580	0.761	0.177	0.316	0.569
CK	0.753	0.268	0.400	0.586	0.770	0.259	0.389	0.570	0.750	0.273	0.397	0.583
CKCC- <i>l</i>	0.733	0.182	0.324	0.560	0.743	0.180	0.313	0.558	0.739	0.172	0.310	0.563
CKCC	0.735	0.181	0.323	0.562	0.728	0.175	0.335	0.571	0.740	0.169	0.318	0.553

Table 5: Performance Comparison for All Methods on FTF students on Different Majors

Method	MATH				PHYS				CHEM			
	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂
MF	0.762	0.234	0.336	0.523	1.099	0.106	0.206	0.383	0.684	0.262	0.399	0.601
MFCC- <i>l</i>	0.758	0.195	0.333	0.568	0.960	0.113	0.213	0.447	0.678	0.221	0.374	0.589
MFCC	0.758	0.206	0.322	0.559	0.998	0.163	0.248	0.433	0.663	0.249	0.380	0.592
CK	0.782	0.267	0.378	0.569	0.910	0.135	0.270	0.468	0.680	0.249	0.393	0.595
CKCC- <i>l</i>	0.784	0.184	0.316	0.535	0.978	0.238	0.294	0.437	0.734	0.312	0.449	0.611
CKCC	0.842	0.309	0.413	0.562	0.842	0.254	0.373	0.508	0.697	0.290	0.411	0.620

Method	IT				CS				BIOL			
	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂
MF	0.655	0.201	0.36	0.623	0.723	0.190	0.346	0.595	0.687	0.253	0.411	0.626
MFCC- <i>l</i>	0.664	0.181	0.365	0.630	0.715	0.177	0.326	0.603	0.777	0.317	0.439	0.599
MFCC	0.627	0.231	0.381	0.659	0.704	0.209	0.362	0.605	0.676	0.274	0.429	0.638
CK	0.606	0.299	0.466	0.681	0.722	0.244	0.395	0.597	0.643	0.316	0.464	0.653
CKCC- <i>l</i>	0.693	0.288	0.460	0.632	0.784	0.242	0.376	0.578	0.771	0.341	0.461	0.605
CKCC	0.600	0.310	0.465	0.692	0.696	0.256	0.395	0.612	0.660	0.329	0.467	0.649

Table 6: Performance Comparison for All Methods on TR students on Different Majors

Method	MATH				PHYS				CHEM			
	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂
MF	0.608	0.270	0.433	0.617	0.675	0.235	0.431	0.569	0.749	0.219	0.325	0.553
MFCC- <i>l</i>	0.637	0.270	0.418	0.610	0.669	0.216	0.353	0.588	0.634	0.281	0.412	0.649
MFCC	0.621	0.241	0.397	0.645	0.577	0.353	0.471	0.667	0.675	0.228	0.404	0.649
CK	0.573	0.394	0.545	0.677	0.741	0.200	0.275	0.550	0.679	0.368	0.491	0.623
CKCC- <i>l</i>	0.641	0.384	0.515	0.677	0.694	0.325	0.450	0.625	0.651	0.377	0.500	0.667
CKCC	0.613	0.404	0.576	0.707	0.805	0.200	0.350	0.600	0.642	0.404	0.518	0.675

Method	IT				CS				BIOL			
	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂	MAE	PTA ₀	PTA ₁	PTA ₂
MF	0.614	0.217	0.405	0.662	0.836	0.175	0.302	0.538	0.711	0.200	0.341	0.559
MFCC- <i>l</i>	0.610	0.227	0.419	0.665	0.818	0.189	0.325	0.541	0.670	0.213	0.366	0.617
MFCC	0.608	0.243	0.415	0.658	0.796	0.193	0.333	0.578	0.674	0.206	0.367	0.604
CK	0.608	0.223	0.406	0.659	0.737	0.212	0.369	0.577	0.695	0.226	0.370	0.600
CKCC- <i>l</i>	0.598	0.235	0.426	0.659	0.756	0.184	0.343	0.599	0.679	0.228	0.384	0.600
CKCC	0.602	0.231	0.412	0.672	0.773	0.234	0.371	0.563	0.643	0.260	0.393	0.629

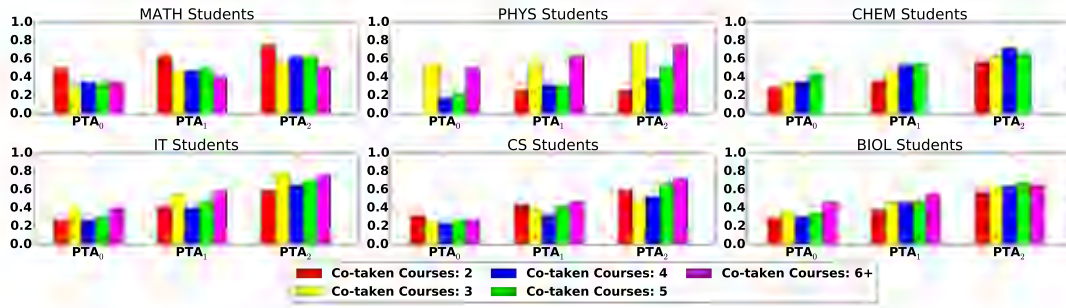


Figure 4: PTA Results for Different Number of Co-taken Courses on FTF students

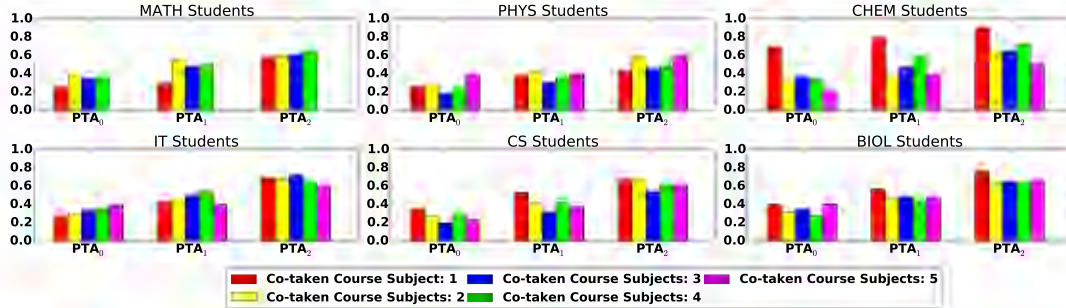


Figure 5: PTA Results for Different Number of Co-taken Course Subjects on FTF students

co-taken courses. This observation suggests that CKCC is able to leverage stronger influence of co-taken courses to improve its performance. However, for PHYS and CS majors, CKCC achieves better performance with 2, 3 or 6+ co-taken courses than with 4 or 5 co-taken courses. We postulate that this is due to the characteristics of courses chosen within a term and their content. These results also indicate that CKCC is able to model co-taken courses' influence despite of the number of the co-taken courses.

6.5 Performance on Different Numbers of Co-taken Course Subjects

In this section, we extract each course's subject and test the CKCC model on different data subgroups with different number of co-taken course subjects in a term. The reason we conduct this experiment is because we assume that courses with the same subject tend to have relevant knowledge components. Students who have co-taken courses from many different subjects may have wide knowledge diversity. This experiment aims to test the performance of CKCC in terms of co-taken course subjects.

Specifically, we take the students in the test set and divide them into five groups: students who take courses from $\{1, 2, 3, 4, 5\}$ subjects in a term. Since there are few students co-taking courses from 6+ subjects, we exclude these students in our experiment. We perform this group of experiment on each major for both FTF and TR students, respectively. For the sake of page limit, we only show the results for FTF students. Figure 5 shows the experimental results in terms of PTA_0 , PTA_1 and PTA_2 . The results show that CKCC have different prediction results regarding the number of co-taken course subjects for different majors. For example, for CHEM, CS and BIOL majors, the performance of the CKCC model on PTA has the best performance with 1 co-taken course subject than other

subgroups. This observation suggests that CKCC is able to model co-taken courses' influence better with less knowledge diversity in a term. However, for IT major, CKCC achieves better performance with more co-taken course subjects. And for MATH and PHYS majors, CKCC has better performance on 2 or 5 co-taken course subjects than other subgroups. We assume that this is affected by the characteristics of different majors. Moreover, for MATH and IT major, the PTA results don't vary much comparing to CHEM and BIOL majors. This illustrates that for some majors, students may take courses from several subjects at a term, and the CKCC model can still well capture the co-taken courses' influence.

7. SIGNIFICANCE AND IMPACT

To highlight the use-case scenario of the developed next term grade prediction approach using co-taken courses, we ran a simulated case study. Having demonstrated the prediction accuracy of these proposed models, the objective of this case study is to highlight the strengths of the proposed models in helping students to select courses in the future term. Implicitly we want to provide students information about their workload (or change in their overall grades) by addition of one or more courses within the next term.

Specifically, we extract two pairs of popular co-taken courses: BIOL311 ("General Genetics") and CHEM313 ("Organic Chemistry"), MATH213 ("Analytic Geometry and Calculus II") and PHYS260 ("University Physics"), and conduct a study to illustrate how our model can help plan students' course selections or allocate the necessary study time. Take the course pair BIOL311 and CHEM313 as an example. We extract the students who take course BIOL311 and CHEM313 together in a term. We predict students' performance on course BIOL311 using the CKCC model. We then eliminate course CHEM313 from our data set and predict the grade on course BIOL311 again using the CKCC model. Comparing the

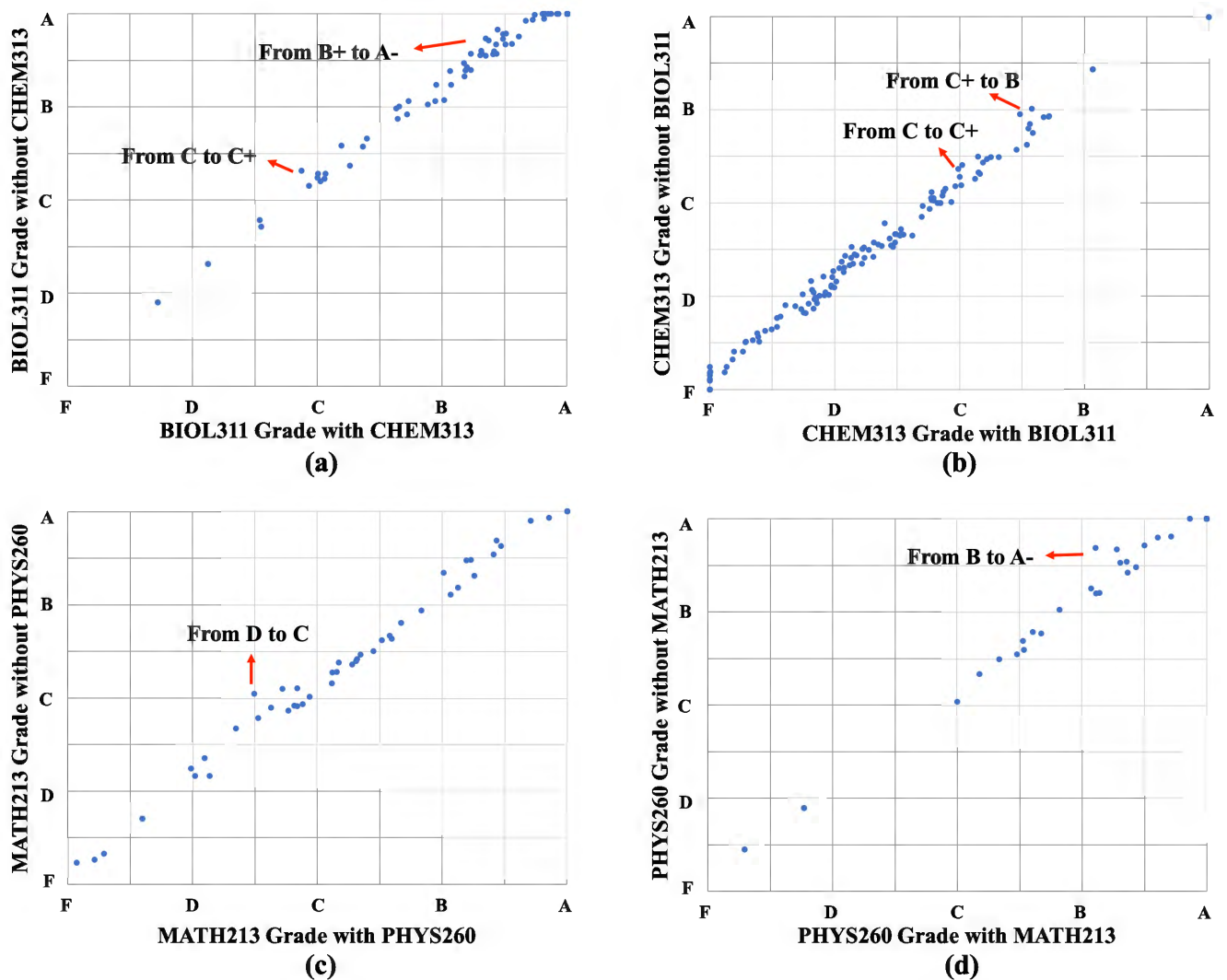


Figure 6: Comparison Results on the Co-taken Course Influence

predicted grades helps determine if the two courses should be taken together within the same term or not. The sampled students have a total of five courses that they are enrolled in for the particular term. The comparison results are shown in Figure 6 (a). It is a scatter plot of predicted grades for a student where the x-axis shows the performance on course BIOL311 co-taken with the CHEM313 and the y-axis is the performance on course BIOL311 with course CHEM313 removed. We have conducted the same experiments for other course pairs using the same protocol and shown these results in Figure 6 (b), (c) and (d).

In general, students' performance will get better with the other course eliminated due to the reduction in workload. However, different students get affected differently by the additional course. For students who take BIOL311 and CHEM313, some of them will have improvement in BIOL311 grades if they do not enroll for CHEM313 in the same semester. On the other hand, some students will not have any change in their grades for BIOL311 based on course CHEM313 (the plotted results along the diagonal). Similar trends can be observed in Figure 6 (b), (c) and (d) as well. In

the Figure 6, we also highlight different cases where students grade changes with the removal of the particular course. Using this information, students can plan the set of courses that they might enroll for in the next term, and allocate study time accordingly.

8. CONCLUSION AND FUTURE WORK

In this work, we propose grade prediction models that incorporate both cumulative knowledge and co-taken courses (CKCC) to predict students' performance in the next term. The proposed models consider both cumulative knowledge a student has acquired after taking a series of courses in the passing terms, and the co-taken courses the student plans to take in the next term. Our experimental results on a dataset from George Mason University shows that the proposed models significantly outperform other competitive baselines over most the datasets for the task of next-term grade prediction. Moreover, our experimental results show that the proposed model is able to capture strong influence of co-taken courses to improve its grade prediction performance. Furthermore, we ran a simulated case study to illustrate how our proposed model can help students in course selection for the future term.

In the future, we plan to take into account additive factors, such as instructor, student's academic level and course's difficulty level along with co-taken course information, in order to achieve more accurate grade prediction results. We hope such a grade prediction system can not only help students select courses, finish their study at college but also guide them in career planning in the future.

9. ACKNOWLEDGMENTS

Funding was provided by NSF Grant, 1447489.

10. REFERENCES

- [1] Asmaa Elbadrawy and George Karypis. Domain-aware grade prediction and top-n course recommendation. *Boston, MA, Sep*, 2016.
- [2] Asmaa Elbadrawy, Agoritsa Polyzou, Zhiyun Ren, Mackenzie Sweeney, George Karypis, and Huzefa Rangwala. Predicting student performance using personalized analytics. *Computer*, 49(4):61–69, 2016.
- [3] Asmaa Elbadrawy, Scott Studham, and George Karypis. Personalized multi-regression models for predicting students performance in course activities. *UMN CS 14-011*, 2014.
- [4] Chein-Shung Hwang and Yi-Ching Su. Unified clustering locality preserving matrix factorization for student performance prediction. *IAENG Int. J. Comput. Sci*, 42(3):245–253, 2015.
- [5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [6] Severin Klingler, Rafael Wampfler, Tanja Käser, Barbara Solenthaler, and Markus Gross. Efficient feature embeddings for student classification with variational auto-encoders.
- [7] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.
- [8] Sara Morsy and George Karypis. Cumulative knowledge-based regression models for next-term grade prediction. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 552–560. SIAM, 2017.
- [9] Michelle Parker. Advising for retention and graduation. 2015.
- [10] Štefan Pero and Tomáš Horváth. Comparison of collaborative-filtering techniques for small-scale student performance prediction task. In *Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering*, pages 111–116. Springer, 2015.
- [11] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.
- [12] Agoritsa Polyzou and George Karypis. Grade prediction with models specific to students and courses. *International Journal of Data Science and Analytics*, pages 1–13, 2016.
- [13] Zhiyun Ren, Xia Ning, and Huzefa Rangwala. Grade prediction with temporal course-wise influence. *arXiv preprint arXiv:1709.05433*, 2017.
- [14] Zhiyun Ren, Xia Ning, and Huzefa Rangwala. Ale: Additive latent effect models for grade prediction. *arXiv preprint arXiv:1801.05535*, 2018.
- [15] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor. Recommender systems handbook., 2011.
- [16] Arjun Sharma, Arijit Biswas, Ankit Gandhi, Sonal Patil, and Om Deshmukh. Livelinet: A multimodal deep recurrent neural network to predict liveliness in educational videos. In *EDM*, pages 215–222, 2016.
- [17] Jill M Simons. *A National Study of Student Early Alert Models at Four-Year Institutions of Higher Education*. ERIC, 2011.
- [18] Mack Sweeney, Jaime Lester, and Huzefa Rangwala. Next-term student grade prediction. In *Big Data (Big Data), 2015 IEEE International Conference on*, pages 970–975. IEEE, 2015.
- [19] Mack Sweeney, Huzefa Rangwala, Jaime Lester, and Aditya Johri. Next-term student performance prediction: A recommender systems approach. *arXiv preprint arXiv:1604.01840*, 2016.
- [20] Nguyen Thai-Nghe, Lucas Drumond, Artus Krohn-Grimberghe, and Lars Schmidt-Thieme. Recommender system for predicting student performance. *Procedia Computer Science*, 1(2):2811–2819, 2010.
- [21] Andreas Zell. *Simulation neuronaler netze*, volume 1. Addison-Wesley Bonn, 1994.

Leveraging Deep Reinforcement Learning for Pedagogical Policy Induction in an Intelligent Tutoring System

Markel Sanz Ausin, Hamoon Azizsoltani, Tiffany Barnes and Min Chi
North Carolina State University
Raleigh, NC, 27695
{msanzau,hazizso,tmbarnes,mchi}@ncsu.edu

ABSTRACT

Deep Reinforcement Learning (DRL) has been shown to be a very powerful technique in recent years on a wide range of applications. Much of the prior DRL work took the *online* learning approach. However, given the challenges of building accurate simulations for modeling student learning, we investigated applying DRL to induce a pedagogical policy through an *offline* approach. In this work, we explored the effectiveness of offline DRL for pedagogical policy induction in an Intelligent Tutoring System. Generally speaking, when applying offline DRL, we face two major challenges: one is limited training data and the other is the credit assignment problem caused by delayed rewards. In this work, we used Gaussian Processes to solve the credit assignment problem by estimating the inferred immediate rewards from the final delayed rewards. We then applied the DQN and Double-DQN algorithms to induce adaptive pedagogical strategies tailored to individual students. Our empirical results show that without solving the credit assignment problem, the DQN policy, although better than Double-DQN, was no better than a random policy. However, when combining DQN with the inferred rewards, our best DQN policy can outperform the random yet reasonable policy, especially for students with high pre-test scores.

1. INTRODUCTION

Interactive e-Learning Environments such as Intelligent Tutoring Systems (ITSs) and educational games have become increasingly prevalent in educational settings. In order to design effective interactive learning environments, developers must form the basic core of the system and determine *what* is to be taught and *how*. *Pedagogical strategies* are policies that are used to decide the *how* part, what action to take next in the face of alternatives. Each of these systems' decisions will affect the user's subsequent actions and performance.

Reinforcement Learning (RL) is one of the best machine learning approaches for decision making in interactive envi-

ronments and RL algorithms are designed to induce effective policies that determine the best action for an agent to take in any given situation to maximize some predefined cumulative reward. In recent years, deep neural networks have enabled significant progress in RL research. For example, Deep Q-Networks (DQNs) [26] have successfully learned to play Atari games at or exceeding human level performance by combining deep convolutional neural networks and Q-learning. Since then, DRL has achieved notable successes in a variety of complex tasks such as robotics control [1] and the game of Go [44]. From DQN, various DRL methods such as Double DQN [51] or Actor-Critic methods [38, 39] were proposed and shown to be more effective than the classic DQN. Despite DRL's great success, there are still many challenges preventing DRL from being applied more broadly in practice, including applying it to educational systems. One major problem is sample inefficiency of current DRL algorithms. For example, it takes DQN hundreds of millions of interactions with the environment to learn a good policy and generalize to unseen states, while we seek to learn policies from datasets with fewer than 800 student-tutor interaction logs.

Generally speaking, there are two major categories of RL: online and offline. Online RL algorithms learn policy while the agent interacts with the environment; offline RL algorithms, by contrast, learn the policy from pre-collected training data. Online RL methods are generally appropriate for domains where the state representation is clear and interacting with simulations and actual environments is relatively computationally cheap and feasible, so most of prior work on DRL mainly took an online learning approach. On the other hand, for domains such as e-learning, building accurate simulations or simulating students is especially challenging because human learning is a rather complex, not fully understood process; moreover, learning policies while interacting with students may not be feasible and more importantly, may not be ethical. Therefore, our DRL approach is offline. This approach was achieved by, first, collecting a training corpus. One common convention, and the one used in our study, is to collect an *exploratory* corpus by training a group of students on an ITS that makes random yet *reasonable* decisions and then apply RL to induce pedagogical policies from that exploratory training corpus. An empirical study was then conducted from a new group of human subjects interacting with different versions of the system. The only difference among the versions was the policy employed by the ITS. Lastly, the students' performance was statistically

Markel Sanz Ausin, Hamoon Azizsoltani, Tiffany Barnes and Min Chi "Leveraging Deep Reinforcement Learning for Pedagogical Policy Induction in an Intelligent Tutoring System" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 168 - 177

compared. Due to cost limitations, typically, only the *best* RL-induced policy was deployed and compared against some baseline policies.

When applying offline DRL to ITSs, we often face one major challenge: our rewards are often not only noisy but also delayed. Given the nature of ITS data collection, the training data including our reward functions is often noisy and our rewards are only the incomplete or imperfect observations of underlying true reward mechanisms. Due to the complex nature of student learning, the most appropriate rewards are (delayed) student learning gains, which are only available after the entire training is complete. For example, hints might improve immediate performance but negatively impact overall learning. On the other hand, when the size of the training data is limited, the availability of “true” immediate rewards is very important for offline RL. Immediate rewards are generally more effective than delayed rewards for offline RL because it is easier to assign appropriate credit or blame when the feedback is tied to a single decision. The more we delay rewards or punishments, the harder it becomes to assign credit or blame properly. Therefore, the challenge is how to distribute the delayed rewards to observable, immediate rewards along each student-system interactive trajectory while taking the noise and uncertainty in the data into account. To tackle this issue, we applied a Gaussian Processes based (GP-based) approach to infer “immediate rewards” from the delayed rewards and then applied DQN to induce two policies: one based on delayed rewards and the other based on the inferred immediate rewards, referred to as DQN-Del and DQN-Inf respectively.

In this work, we used a logic ITS and focused on applying DRL to induce a policy on one type of tutorial decision: whether to present a given problem as a *problem solving* (PS) or a *worked example* (WE). The tutor presents a worked example (WE) by demonstrating the individual steps in an expert solution to a problem. During PS, students are required to complete the problem with tutor support (e.g. hints). The effectiveness of DQN-Del and DQN-Inf are evaluated theoretically using Expected Cumulative Reward (ECR) and empirically through two randomly controlled experiments: one for evaluating the effectiveness of DQN-Del in Spring 2018 and the other for evaluating DQN-Inf in Fall 2018. In each experiment, the effectiveness of the corresponding RL-induced policy was compared against the *Random* policy that flips a coin to decide between WE/PS and the students were randomly assigned into the two conditions while balancing their incoming competence. Overall, the results from both experiments showed no significant difference between the DQN-Del and Random in Spring 2018 and between the DQN-Inf and Random in Fall 2018 on every measure of learning performance.

There are two potential explanations for such findings. First, our random baseline policy is decently strong. While random policies are usually bad in many RL tasks, in the context of WE vs. PS, our random policies can be strong baselines. Indeed, some learning literature suggests that the best instructional intervention is to alternate WE and PS [35, 41, 36]. Second, there may be an **aptitude-treatment interaction** (ATI) effect [6, 47], where certain students are less sensitive to the induced policies, meaning they achieve a

similar learning performance regardless of policies employed; whereas other students are more sensitive, meaning their learning is highly dependent on the effectiveness of the policies. Thus, we divided the students into High vs. Low based on their incoming competence and investigated the ATI effect. While no ATI effect was found between DQN-Del and Random for Spring 2018, a significant ATI effect was found between DQN-Inf and Random in Fall 2018.

In short, we explored applying offline DRL for pedagogical policy induction based on delayed and inferred immediate rewards. Our results showed that no ATI effect was found between DQN-Del and Random in Spring 2018, whereas there was an ATI effect between DQN-Inf and Random in Fall 2018. More specifically, the High incoming competence group benefited significantly more from the DQN-Inf policy than their peers in the Random condition. This result suggests that the availability of inferred immediate rewards was crucial for effectively applying offline DRL for pedagogical policy induction.

2. BACKGROUND

A great deal of research has investigated the differing impacts of worked examples (WE) and problem solving (PS) on student learning [49, 22, 21, 23, 41, 27, 36]. McLaren and colleagues compared WE-PS pairs with PS-only [22]. Every student was given a total of 10 training problems. Students in the PS-only condition were required to solve every problem while students in the WE-PS condition were given 5 example-problem pairs. Each pair consisted of an initial worked example problem followed by tutored problem solving. They found no significant difference in learning performance between the two conditions. However, the WE-PS group spent significantly less time than the PS group.

McLaren and his colleagues found similar results in two subsequent studies [21, 23]. In the former, the authors compared three conditions: WE, PS and WE-PS pairs, in the domain of high school chemistry. All students were given 10 identical problems. As before, the authors found no significant differences among the three groups in terms of learning gains but the WE group spent significantly less time than the other two conditions; and no significant time on task difference was found between the PS and WE-PS conditions.

In a follow-up study, conducted in the domain of high school stoichiometry, McLaren and colleagues compared four conditions: WE, tutored PS, untutored PS, and Erroneous Examples (EE) [23]. Students in the EE condition were given *incorrect* worked examples containing between 1 and 4 errors and were tasked with correcting them. The authors found no significant differences among the conditions in terms of learning gains, and as before the WE students spent significantly less time than the other groups. More specifically, for time on task, they found that: $WE < EE < untutored PS < tutored PS$. In fact, the WE students spent only 30% of the total time that the tutored PS students spent.

The advantages of WEs were also demonstrated in another study in the domain of electrical circuits [50]. The authors of that study compared four conditions: WE, WE-PS pairs, PS-WE pairs (problem-solving followed by an example problem), and PS only. They found that the WE and WE-PS

students significantly outperformed the other two groups, and no significant differences were found among four conditions in terms of time on task.

In short, prior research has shown that WE can be similar or more effective than PS or alternating PS with WE, and the former can take significantly less time than the latter two [49, 22, 21, 23, 41]. However, there is no widespread consensus on how or when WE vs. PS should be used. This is why we will derive pedagogical strategies for them directly from empirical data.

2.1 ATI Effect

Previous work shows that the ATI effect commonly exists in many real-world studies. More formally, the ATI effect states that instructional treatments are more or less effective to individual learners depending on their abilities [6]. For example, Kalyuga et al. [17] empirically evaluated the effectiveness of worked example (WE) vs. problem solving (PS) on student learning in programmable logic. Their results show that WE is more effective for inexperienced students while PS is more effective for experienced learners.

Moreover, D’Mello et al. [7] compared two versions of ITSs: one is an affect-sensitive tutor which selects the next problem based on students’ affective and cognitive states combined, while the other is an original tutor which selects the next problem based on students’ cognitive states alone. An empirical study shows that there is no significant difference between the two tutors for students with high prior knowledge. However, there is a significant difference for students with low prior knowledge: those who trained on the affect-sensitive tutor had significantly higher learning gain than their peers using the original tutor.

Chi and VanLehn [4] investigated the ATI effect in the domain of probability and physics, and their results showed that high competence students can learn regardless of instructional interventions, while for students with low competence, those who follow the effective instructional interventions learned significantly more than those who did not. Shen and Chi [43] find that for pedagogical decisions on WE vs. PS, certain learners are always less sensitive in that their learning is not affected, while others are more sensitive to variations in different policies. In their study, they divided students into Fast and Slow groups based on time, and found that the Slow groups are more sensitive to the pedagogical decisions while the Fast groups are less sensitive.

3. RELATED WORK

Deep Reinforcement Learning: In recent years, many DRL algorithms have been developed for various applications such as board games like Go [44, 46], Chess and Shogi [45], robotic hand dexterity [33, 1], physics simulators [19, 29, 30], and so forth. While most DRL algorithms have been mainly applied online, some of them can also be applied offline. More specifically, DRL algorithms such as Vanilla Policy Gradient (VPG) [48], Proximal Policy Optimization (PPO) [39], Trust Region Policy Optimization (TRPO) [38], or A3C [24] can only be applied for online learning by interacting with simulations. Some other DRL algorithms can be applied for offline learning using pre-collected training data. These include the Q-learning based approaches such as Deep

Q-Network (DQN) [26], Double-DQN [51], prioritized experience replay [37], distributed prioritized experience replay (Ape-X DQN) [14], and the Actor-Critic based methods such as Deep Deterministic Policy Gradient (DDPG) [19], Twin Delayed Deep Deterministic policy gradient (TD3) [9], or Soft Actor-Critic (SAC) [11]. Among them, DQN and its variants have been much more extensively studied, however, it is still not clear whether they can be successfully applied *offline* for pedagogical policy induction for ITSs.

Reinforcement Learning in Education: Prior research using online RL to induce pedagogical policies has often relied on simulations or simulated students, and the success of RL is often heavily dependent on the accuracy of the simulations. Beck et al. [3] applied temporal difference learning, with off-policy ϵ -greedy exploration, to induce pedagogical policies that would minimize student time on task. Iglesias et al. applied another common online approach named Q-learning to induce policies for efficient learning [15, 16]. More recently, Rafferty et al. applied POMDP with tree search to induce policies for faster learning [32]. Wang et al. applied an online Deep-RL approach to induce a policy for adaptive narrative generation in educational game [52]. All of the models described above were evaluated by comparing the induced policy with some baseline policies via simulations or classroom studies.

Offline RL approaches, on the other hand, “take advantage of previously collected samples, and generally provide robust convergence guarantees” [40]. Shen et al. applied value iteration and least square policy iteration on a pre-collected training corpus to induce pedagogical policies for improving students’ learning performance [43, 42]. Chi et al. applied policy iteration to induce a pedagogical policy aimed at improving students’ learning gains [5]. Mandel et al. [20] applied an offline POMDP approach to induce a policy which aims to improve student performance in an educational game. In classroom studies, most models above were found to yield certain improved student learning relative to a baseline policy.

DRL in Education is a subject of growing interest. DRL adds deep neural networks to RL frameworks such as POMDP for function approximation or state approximation [25, 26]. This enhancement makes the agent capable of achieving complicated tasks. Wang et al. [52] applied a DRL framework for personalizing interactive narratives in an educational game called CRYSTAL ISLAND. They designed the immediate rewards based on normalized learning gain (NLG) and found that the students with the DRL policy achieved a higher NLG score than those following the linear RL model in *simulation* studies. Furthermore, Narasimhan et al. [28] implemented a Deep Q-Network (DQN) approach in text-based strategy games, constructed based on Evennia, which is an open-source library and toolkit for building multi-users online text-based games. Using simulations, they found that the DRL policy significantly outperformed the random policy in terms of quest completion.

In summary, compared with MDP and POMDP, relatively little research has been done on successfully applying DRL to the field of ITS. None of the prior research has successfully applied DRL to ITSs without simulated environments,

in order to learn an effective pedagogical strategy that makes students learn in a more efficient manner. Furthermore, no prior work has empirically evaluated any DRL-induced policy to confirm its benefits on real students.

4. METHODS

In RL, the agent interacts with an environment \mathcal{E} , and the goal of the agent is to learn a policy that will maximize the sum of future discounted rewards (also known as the return) along the trajectories, where each trajectory is one run through the environment, starting in an initial state and ending in a final state. This is done by learning which action to take for each possible state. In our case, \mathcal{E} is the learning context, and the agent must learn to take the actions that lead to the optimal student learning, by maximizing the return $R = \sum_{t=0}^T \gamma^t r_t$, where r_t is the reward at time step t , T is the time step that indicates the end of the trajectory, and $\gamma \in (0, 1]$ is the discount factor.

4.1 DQN and Double-DQN

Deep Q-Network (DQN) is, fundamentally, a version of Q-learning. In Q-learning, the goal is to learn the optimal action-value function, $Q^*(s, a)$, which is defined as the expected reward obtained when taking the optimal action a in state s , and following the optimal policy π^* until the end of the trajectory. For any state-action pair, the optimal action-value function must follow the Bellman optimality equation in that:

$$Q^*(s, a) = r + \gamma \max_{a'} Q^*(s', a') \quad (1)$$

Here r is the expected immediate reward for taking action a at state s ; γ is the discount factor; and $Q^*(s', a')$ is the optimal action-value function for taking action a' at the subsequent state s' and following policy π^* thereafter.

Compared with the original Q-learning, DQNs use neural networks (NNs) to approximate action-value functions. This is because NNs are great universal function approximators and they are able to handle continuous values in both their inputs and outputs. In order to train the DQN algorithm, two neural networks with equal architectures are employed. One is the main network and its weights are denoted θ and the other is the target network, and its weights are denoted θ^- . The target value used to train the network is $y := r + \gamma \max_{a'} Q(s', a'; \theta^-)$. Thus, the loss function that is minimized in order to train the main network is:

$$Loss(\theta) = \mathbb{E}[(y - Q(s, a; \theta))^2] \quad (2)$$

The main network is trained on every training iteration, while the target network is frozen for a number of training iterations. Every m training iterations, the weights of the main neural network are copied into the target network. This is one of the techniques used in order to avoid divergence during the training process. Another one of these techniques was the use of an experience replay buffer. This buffer contains the p most recent (s, a, r) tuples, and the algorithm randomly samples from the buffer when creating the batch on each training iteration. We followed the same procedure, but as our training was performed offline, the experience replay buffer consists of all the samples on our training corpus, and it does not get refreshed over time.

Double-DQN or DDQN was proposed by Van Hasselt et al. [12] who combined it with neural networks in the Double-DQN algorithm [51]. The intuition behind it is to decouple the action selection from the action evaluation. To achieve this, the Double-DQN algorithm uses the main neural network to first select the action that has the highest Q-value for the next state ($\arg\max_{a'} Q(s', a'; \theta)$) and then evaluates the Q-value of the selected action using the target network ($Q(s', \arg\max_{a'} Q(s', a'; \theta); \theta^-)$). This simple trick has been proven to significantly reduce overestimations in Q-value calculations, resulting in better final policies. With this technique, the target value used to optimize the main network becomes:

$$y := r + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta); \theta^-) \quad (3)$$

The loss function is still the same as in equation 2, but the target value y used in the formula is now updated to be the one in equation 3.

4.2 Fully Connected vs. LSTM

For our NN architectures, we explored two options: Fully connected NNs and Long Short Term Memory (LSTM).

Fully Connected or multi-layer perceptrons are the simplest form of neural network units. They calculate a simple weighted sum of all the input units, and each unit produces an output value that is often passed to an activation function. We used these units to parametrize our neural networks. All the input units are connected to all the units in the first hidden layer, and all those units are connected to every unit in the next hidden layer. This process continues until the final output layer.

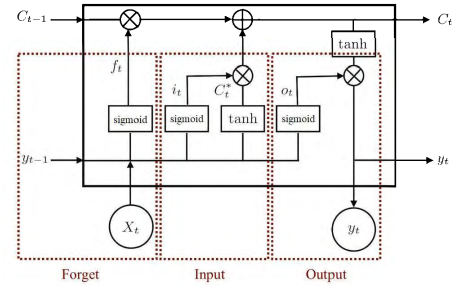


Figure 1: A single LSTM unit containing a forget, input and output gate

Long Short-Term Memory (LSTM) is a type of recurrent neural network specifically designed to avoid the vanishing and exploding gradient problems [13]. LSTMs are particularly suitable for tasks where long-term temporal dependencies must be remembered. They achieve this by maintaining the previous information of hidden states as internal memory. Figure 1 shows the architecture of a single LSTM unit. It consists of a memory cell state denoted by C_t and three gates: the forget gate $f_t \in [0, 1]$, the input gate $i_t \in [0, 1]$, and the output gate $o_t \in [0, 1]$. These three gates interact with each other to control the flow of information. During training, the network learns what to memorize and when to allow writing to the cell in order to minimize the

training error. More specifically, the forget gate determines what information from the previous memory cell state is expired and should be removed; the input gate selects information from the *candidate* memory cell state C_t^* to update the cell state; and the output gate filters the information from the memory cell so that the model only considers information relevant to the prediction task. The value of each gate is computed as follows, where $W_{[i,f,C,o]}$ are the weight matrices and $b_{[i,f,C,o]}$ are the bias vectors:

$$\begin{aligned} i_t &= \text{sigmoid}(W_i \cdot [y_{t-1}, X_t] + b_i) \\ f_t &= \text{sigmoid}(W_f \cdot [y_{t-1}, X_t] + b_f) \\ C_t^* &= \tanh(W_C \cdot [y_{t-1}, X_t] + b_c) \\ o_t &= \text{sigmoid}(W_o \cdot [y_{t-1}, X_t] + b_o) \end{aligned} \quad (4)$$

The memory cell value C_t and output value y_t from the LSTM unit are computed using the following formulas:

$$\begin{aligned} C_t &= C_{t-1} \cdot f_t + C_t^* \cdot i_t \\ y_t &= o_t * \tanh(C_t) \end{aligned} \quad (5)$$

4.3 Inferring Immediate Rewards

A historical dataset \mathcal{H} consists of m trajectories, h_1 to h_m and n unknown immediate rewards. We would like to infer the immediate rewards given delayed rewards. In order to infer the immediate rewards, we used a minimum mean square error (MMSE) estimator in the Bayesian setting [18, 8, 10]. Assume $\mathbf{R} = \mathbf{D}\mathbf{r} + \varepsilon$ is a linear process where \mathbf{D} is a known matrix, \mathbf{r} is a $n \times 1$ random vector of unknown immediate rewards, \mathbf{R} is a $m \times 1$ vector of observed delayed rewards and ε is a vector of independent and identically distributed noise with mean of zero and standard deviation of $\sigma_{\mathbf{R}}$. Assuming the discounted sum of the immediate rewards is equal to the delayed rewards, a linear model matrix \mathbf{D} is proposed as:

$$\mathbf{D} = \begin{bmatrix} \overbrace{1 \quad \gamma \quad \gamma^2 \quad \dots}^{h_1} & \overbrace{0 \quad 1 \quad \gamma \quad \gamma^2 \quad \dots}^{h_2} & \dots & 0 \\ 0 & \dots & 0 & 1 & \gamma & \gamma^2 & \dots & 0 & \dots \\ 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \ddots \end{bmatrix} \quad (6)$$

where γ is the discount factor. Following the linear MMSE estimator, we assume that the immediate rewards follow a Gaussian Process defined as $\mathbf{r} \sim \mathcal{N}(\mu_{\mathbf{r}}, \mathbf{C}_{\mathbf{rr}})$ where $\mu_{\mathbf{r}}$ is the a priori mean and $\mathbf{C}_{\mathbf{rr}}$ is the a priori covariance defined by an appropriate kernel [2]. Using the theorem of conditional distribution of multivariate Gaussian distributions [34], conditional expectation of immediate rewards given delayed rewards $\mathbb{E}[\mathbf{r}|\mathbf{R}]$ or the posterior mean of immediate rewards is:

$$\mathbb{E}[\mathbf{r}|\mathbf{R}] = \mu_{\mathbf{r}} + \mathbf{C}_{\mathbf{rr}}\mathbf{D}^T\mathbf{C}_{\mathbf{RR}}^{-1}(\mathbf{R} - \mathbf{D}\mu_{\mathbf{r}}) \quad (7)$$

and the posterior covariance $\mathbf{C}[\mathbf{r}|\mathbf{R}]$ of inferred immediate rewards given delayed rewards can be calculated as:

$$\mathbf{C}[\mathbf{r}|\mathbf{R}] = \mathbf{C}_{\mathbf{rr}} - \mathbf{C}_{\mathbf{rr}}\mathbf{D}^T\mathbf{C}_{\mathbf{RR}}^{-1}\mathbf{D}\mathbf{C}_{\mathbf{rr}}^T \quad (8)$$

where $\mathbf{C}_{\mathbf{RR}} = \mathbf{D}\mathbf{C}_{\mathbf{rr}}\mathbf{D}^T + \sigma_{\mathbf{R}}^2\mathbf{I}$ and \mathbf{I} is the identity matrix.

Algorithm 1 shows the process used to infer the immediate rewards. Estimation of the mean and covariance of the ran-

dom column vector \mathbf{r} in Eqs. 7 and 8 requires the inverse of the matrix $\mathbf{C}_{\mathbf{RR}}$. By introducing several intermediary variables, this algorithm provides an efficient solution to matrix inversion using the Cholesky decomposition similar to the Gaussian Processes algorithm implementation [34].

Algorithm 1 Immediate reward approximation algorithm.

Inputs: $\mathbf{R}, \mu_{\mathbf{r}}, \mathbf{C}_{\mathbf{rr}}, \mathbf{D}, \sigma_{\mathbf{R}}^2$
 $\mathcal{L} = \text{Cholesky}(\mathbf{D}\mathbf{C}_{\mathbf{rr}}\mathbf{D}^T + \sigma_{\mathbf{R}}^2\mathbf{I})$
 $\beta = \mathcal{L} \setminus (\mathbf{R} - \mathbf{D}\mu_{\mathbf{r}})$ forward-substitution algorithm
 $\alpha = \mathcal{L}^T \setminus \beta$ back-substitution algorithm
 $\bar{\mathbf{k}} = \mathbf{D}\mathbf{C}_{\mathbf{rr}}^T$
 $\mathbf{v} = \mathcal{L} \setminus \bar{\mathbf{k}}$
 $\mathbb{E}[\mathbf{r}|\mathbf{R}] = \mu_{\mathbf{r}} + \bar{\mathbf{k}}^T \alpha$
 $\mathbf{C}[\mathbf{r}|\mathbf{R}] = \mathbf{C}_{\mathbf{rr}} - \mathbf{v}\mathbf{v}^T$
return: $\mathbb{E}[\mathbf{r}|\mathbf{R}]$ and $\mathbf{C}[\mathbf{r}|\mathbf{R}]$

5. POLICY INDUCTION

In this section, we will describe our ITS, the training corpus, our policy induction procedure, and theoretical evaluation results.

5.1 Logic ITS

The logic tutor used in this study is named Deep Thought (DT), and it uses a graph-based environment to solve logic proofs. It is used in the undergraduate level Discrete Mathematics class at North Carolina State University. To complete a problem, students iteratively apply rules to logic statement nodes in order to derive the conclusion node. DT automatically checks the correctness of each step and provides immediate feedback on any rule that is applied incorrectly. The tutor consists of 6 levels, with 3 to 4 problems per level. Each problem can be represented as Problem Solving (PS) or as Worked Example (WE). Figure 2 (left) shows the user interface for PS, and Figure 2 (right) shows the interface for WE.



Figure 2: User Interface for DT. Left: PS. Right: WE.

5.2 Training Corpus

Our training corpus contains 786 complete student trajectories collected over five semesters. On average, each student spent two hours to complete the tutor. For each student, the tutor makes about 19 decisions. From our student-system interaction logs, we extracted a total of 142 state features:

- **Autonomy:** 10 features describing the amount of work done by the student.
- **Temporal:** 29 features, including average time per step, the total time spent on the current level, the time spent on PS, the time spent on WE, and so on.

- **Problem Solving:** 35 features such as the difficulty of the current problem, the number of easy and difficult problems solved on the current level, the number of PS and WE problems seen in the current level, or the number of nodes the student added in order to reach the final solution.
- **Performance:** 57 features such as the number of incorrect steps, and the ratio of correct to incorrect rule applications for different types of rules.
- **Hints:** 11 features such as the total number of hints requested or the number of hints the tutor provided without the student asking for them.

The features contain non-negative continuous values. As their range varies significantly (time can be a large number while problem difficulty is always between 1 and 9), we normalized each feature to the range $[0, 1]$. Input feature normalization has been shown to improve the stability of the learning process on neural networks, and often leads to faster convergence.

To induce our pedagogical policy, while previous research mainly used learning gains or time on task as reward function, our reward function here is based on the *improvement of learning efficiency*, which balances both learning gain improvement and time on task improvement. In this way, if two students have the same amount of learning gain, the one who takes shorter time would get higher reward. To calculate their learning efficiency, we used students' scores obtained on each level divided by the training time on the level. Students must solve the last problem on each level without help, and we use this as a level score. The range of the score for each level is $[-100, +100]$, and the learning gain for level L is calculated as $Score_L - Score_{L-1}$, thus having a range of $[-200, +200]$.

5.3 Training Process

For both DQN and Double DQN, we explored using Fully Connected (FC) NNs or using LSTM to estimate the action-value function Q . Our FC has four fully connected layers of 128 units each, uses Rectified Linear Unit (ReLU) as the activation function. Our LSTM architecture consists of two layers of 100 LSTM units each, with a fully connected layer at the end. Additionally, for either FC or LSTM, for a given time t , we explored three input settings: to use only the current state observation s_t ($k = 1$), to use the last two state observations: s_{t-1} and s_t ($k = 2$), and to use the last three: s_{t-2} , s_{t-1} and s_t ($k = 3$).

In the case of the fully connected (FC) model, the observations are concatenated and passed to the input layer as a flat array of values. For LSTM, the input state observations are passed to the network in a sequential manner. These past observations provide extra information about the performance of the student in the previous states. However, including previous states also add complexity to the network, which can slow down the learning process and can increase the risk of converging to a weaker final policy. As the number of parameters increases in the NNs, the chance that our NN would get stuck at a local optima increases, especially when our training data is limited. L2 regularization

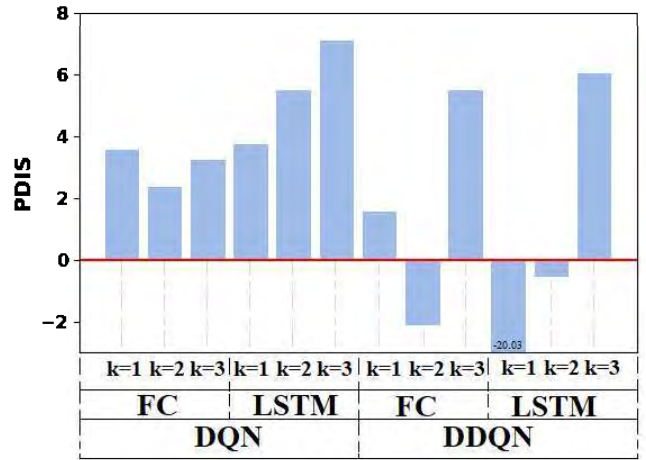


Figure 3: Importance sampling results.

was used to get a model that generalizes better. We trained our models for 50,000 iterations, using a batch size of 200.

5.4 Induced Policy

First, we induced the DQN-Del policy using delayed rewards only. Our training data was split: 90% of the students for training data and 10% for testing data. We trained all 12 of our models (DQN and Double-DQN with either FC layers or LSTM layers, and with $k = \{1, 2, 3\}$) on the training data and evaluated their performance on testing data. We repeated this process twice with two different test sets and reported their average performance on a series of popular off-policy evaluation metrics. Among them, Expected Cumulative Reward (ECR) is the most widely used. However, Per-Decision Importance Sampling (PDIS) has shown to be more robust [31].

ECR is simply calculated by averaging over the highest Q-value for all the initial states in the validation set. The formula is described in Equation 9.

$$ECR = \frac{1}{N} \sum_{i=1}^N \max_a Q(s_i, a) \quad (9)$$

s_i is an initial state, and N denotes the number of trajectories in the validation set.

PDIS [31] is an alternative to regular Importance Sampling, to reduce variance in the estimations. The PDIS results of the 12 models are shown in Figure 3. The PDIS result of the random policy is used to set $y = 0$ (the red line) in Figure 3. Much to our surprise, while double DQN has shown to be much more robust in online DRL applications, its performance is generally worse than DQN here, especially when $k = 1$ and $k = 2$. Figure 3 shows that the best policy is induced using DQN with the LSTM architecture for $k = 3$, and thus is selected as DQN-Del. We also compare the selected policy with the remaining ones using ECR and other evaluation metrics and the results showed using DQN with the LSTM architecture for $k = 3$ is always among the best policies across different evaluation metrics.

To evaluate the impact of Inferred rewards on the DQN in-

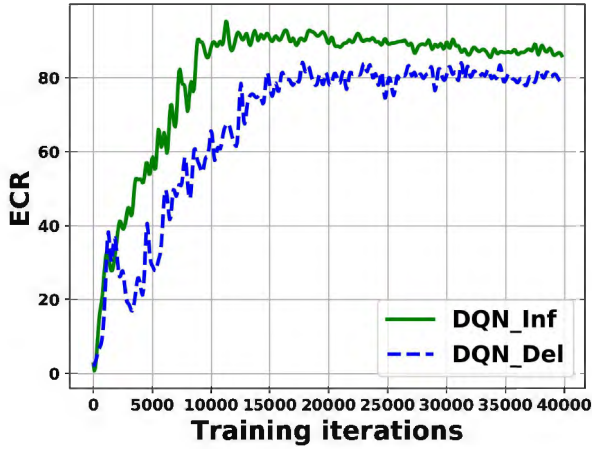


Figure 4: ECR evolution of DQN-Del and DQN-Inf.

duced policies, we used the same approach to induce the DQN-Inf policy and the only major difference is that we used the inferred immediate rewards in the training dataset, calculated through Algorithm 1. During the training process, we calculated the ECRs of DQN with the LSTM architecture for $k = 3$ using the original delayed rewards (DQN-Del) vs. using the inferred immediate rewards (DQN-Inf). The evolution of the ECR values for each policy during the training process is shown in Figure 4, showing that using the inferred rewards we can theoretically converge faster and to a better policy.

6. EMPIRICAL EXPERIMENT SETUP

Two empirical experiments were conducted, one in the Spring 2018 semester and one in the Fall 2018 semester. They were both conducted in the undergraduate Discrete Mathematics class at North Carolina State University.

6.1 Experiment 1: Spring 2018

84 students from the Spring 2018 class were randomly assigned to the *Random* (control) group and the *DQN-Del* group. Because both *WE* and *PS* are considered to be *reasonable* educational interventions in the context of learning, we refer to our control random policy as a *random yet reasonable* policy or *Random* in the following. The assignment was done in a balanced random manner, using the pre-test score to ensure that the two groups had similar prior knowledge. $N = 45$ and $N = 39$ were assigned to *Random* and *DQN-Del* respectively. Among them, $N = 41$ *Random* students and $N = 33$ *DQN-Del* students completed the training. A χ^2 test showed no significant differences between the completion rates of the two different groups: $\chi^2(1, N = 84) = 0.053, p = 0.817$.

6.2 Experiment 2: Fall 2018

98 students from the Fall 2018 Discrete Mathematics class were distributed into two conditions. The two conditions are the *Random* (control) group and the *DQN-Inf* group. The group sizes were as follows: $N = 49$ for *Random*, and $N = 49$ for *DQN-Inf*. A total of 84 students completed the experiment and their distribution was as follows: $N = 43$ for *Random*, and $N = 41$ for *DQN-Inf*. A χ^2 test of inde-

pendence showed no significant differences between the completion rates of the two different groups: $\chi^2(1, N = 98) = 0.025, p = 0.872$.

6.3 Performance Measure

Our tutor is consisting of 6 strictly ordered levels of proof problems. All of the students received the same set of problems in level 1. Their initial proficiency is calculated based upon the number of mistakes made on the final problem of level 1 and the total training time on level 1. The proficiency reflects how well they understand the knowledge and can apply the logic rules in the proof process before the tutor follows different pedagogical policies. In each sequential level, DT will follow the corresponding policies to determine the next problem to be *WE* or *PS*. The last problem on each level is used as a mini-posttest to measure students' performance on that level.

When inducing both the *DQN-Del* and *DQN-Inf*, we calculated our reward function based upon the improvement of students' learning efficiency which is defined as level scores divided by the training time on that level. So to measure student performance, we first calculate the learning efficiency on each level as: the score obtained by the student in the last problem of that level, divided by the total time (in minutes). In this study, we use student learning efficiency in level 1 as their pretest efficiency score and their learning efficiency in level2-level6 as the post-test efficiency scores. Since our DQNs used learning efficiency improvement as their rewards, we expect that the DRL-induced policy would cause students to have higher post-test efficiencies.

7. RESULTS

7.1 Experiment 1 Results

No significant difference was found on the pre-test efficiency between the *Random* and *DQN-Del*: $t(72) = 1.086, p = 0.281$. We divided the students into high pre-test efficiency ($n = 37$) and low pre-test efficiency ($n = 37$) groups, based upon their learning efficiency on the pre-test. As expected, there was a significant difference between the high and low efficiency students on their pre-test efficiency: $t(72) = 9.570, p < 0.001$. The partition mentioned above resulted in four groups, based upon their incoming efficiency and condition: *DQN-Del-High* ($n = 16$), *DQN-Del-Low* ($n = 17$), *Random-High* ($n = 21$), and *Random-Low* ($n = 20$). A t-test showed no significant difference in the pre-test efficiencies either between the two low groups, *Random-Low* and *DQN-Del-Low*, or between the two high efficiency groups. These results show that there is no significant difference in the pre-test efficiency across conditions.

A two-way ANCOVA test on the post-test efficiency, using Condition $\{Random, DQN-Del\}$ and Incoming Competency $\{Low, High\}$ as factors and pre-test efficiency as a covariate, showed that there is no significant main effect of Condition $F(1, 69) = 2.633, p = 0.109$, and no significant main effect of Incoming Efficiency $F(1, 69) = 0.036, p = 0.849$. No interaction (ATI) effect was found either $F(1, 69) = 1.285, p = 0.261$. Thus, we conclude there was no difference between the two conditions in the Spring 2018 study.

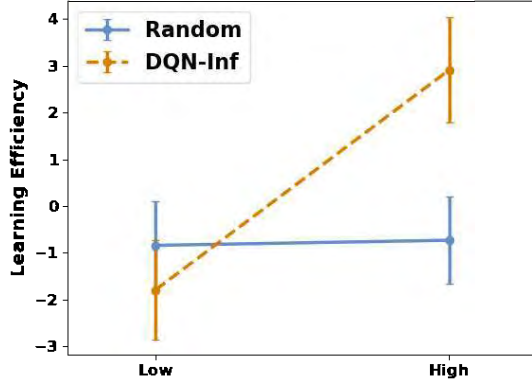


Figure 5: Post-Test Learning Efficiency across different groups for the Fall 2018 study.

7.2 Experiment 2 Results

In fall 2018, again no significant difference was found on the pre-test efficiency between the Random and DQN-Inf groups: $t(82) = -0.333, p = 0.739$. The students were also divided into high pre-test efficiency ($n = 42$) and low pre-test efficiency ($n = 42$) groups. A t-test showed a significant difference between the high and low efficiency students on the pre-test efficiency: $t(82) = 6.38, p < 0.001$. The same four groups were formed, based upon their incoming efficiency and condition: DQN-Inf-High ($n = 20$), DQN-Inf-Low ($n = 21$), Random-High ($n = 22$), and Random-Low ($n = 21$). A t-test showed no significant difference on the pre-test efficiencies when comparing the Random-Low and DQN-Inf-Low groups: $t(40) = 0.027, p = 0.978$. No significant difference was found either, when performing a t-test on the two high efficiency groups: $t(40) = -0.698, p = 0.489$. This shows that there is no significant difference on the pre-test efficiency across conditions during the Fall 2018 study.

A two-way ANOVA test using Condition {*Random*, *DQN-Inf*} and Incoming Competency {*Low*, *High*} as two factors showed a significant interaction effect on students' post-test efficiency: $F(1, 80) = 5.038, p = 0.027$ (as shown in Figure 5). To be more strict, we ran a two-way ANCOVA test using Condition and Incoming Competency as two factors and pre-test efficiency as a covariate. This analysis also showed a significant interaction effect on students' post-test efficiency: $F(1, 79) = 4.687, p = 0.033$. Thus, by taking the pre-test efficiency into consideration, there is still a significant interaction effect. No significant main effect was found from either Condition or Incoming Competency. A one-way ANCOVA test on the post-test efficiency for the Low competency groups, using Condition {*Random-Low*, *DQN-Inf-Low*} as a factor and pre-test competency as a covariate showed no significant difference on the post-test efficiency $F(1, 39) = 0.429, p = 0.516$. However, a significant difference was found for the High groups $F(1, 39) = 5.513, p = 0.024$, with means -0.719 for Random-High and 2.916 for DQN-Inf-High (as shown in Figure 5).

7.3 Log Analysis

This section will show more details on the different types of tutorial decisions made across the different conditions and studies. The features that were analyzed include the total number of problems each student encountered (TotalCount), the number of problems solved (PSCount), the number of difficult problems solved (diffPSCount), the number of WEs seen (WECount), and the number of difficult WEs seen (diffWECount). Table 1 shows the summary of these five features for each condition and study. Columns 3 and 4 show the mean and standard deviation of each condition for these categories. Column 5 shows the statistical results of different t-tests comparing the two conditions.

No significant difference is found for the total number of problems seen by each group. However, we observed that for the features diffPSCount, WECount and diffWECount, a significant difference was found only during the Spring 2018 study. Looking at the mean values, we notice that the DQN-Del policy assigned fewer WE and more PS problems. However, this did not improve the performance of the students in the DQN-Del group during this study. During the Fall 2018 study, we only observe a significant difference in the number of PS problems assigned. No significant difference was found in the remaining categories.

When we analyze the logs for the High competency students, table 2 shows the values of those same features, but only for the High competency students in each study. During the Spring 2018 semester, we find a statistically significant difference for TotalCount, PSCount, and diffWECount, and we find a marginal difference for WECount. This shows that the DQN-Del policy gave more PS problems, fewer WE, and fewer difficult WE problems, but no significant difference was found in students' post-test performance. The Fall 2018 study results show no significant or marginal difference in any of the five categories. Despite this fact, the DQN-Inf policy implemented in the Fall 2018 study outperformed the Random policy for the High competency students. We can also observe how, in Table 2, the standard deviation for the DQN groups is often larger than the standard deviation for the Random groups. This makes sense because we expect all the students in the Random group to have a similar values in each category. However, it looks like the DQN policy is assigning more PS to certain students, and more WE to other students, resulting in a larger standard deviation.

In short, our log analysis results show that it is not about the total amount of PSs and WEs that students received that matters, but rather how or when they receive which.

8. CONCLUSIONS

We used offline Deep Reinforcement Learning algorithms in conjunction with inferred immediate rewards to induce a pedagogical policy to improve the students' learning efficiency for a logic tutor. Our results showed that our DRL-induced pedagogical policy can outperform the Random policy, which is a strong baseline here. More specifically, there was an ATI effect in the Fall 2018 study in that the high incoming competency students were benefited more from our DRL-induced policy, by achieving better post-test learning efficiency than other groups. Our results showed that our proposed Gaussian Processes based approach to infer "im-

Table 1: Log analysis results for per semester and condition.

Feature	Semester	Random	DQN	Significance	
TotalCount	Spring	22.68(5.05)	24.02(5.29)	$t(72) = -1.118$,	$p = 0.267$
	Fall	23.81(3.32)	25.26(5.37)	$t(82) = -1.489$,	$p = 0.141$
PSCount	Spring	14.82(5.29)	17.08(6.11)	$t(72) = -1.691$,	$p = 0.095\bullet$
	Fall	14.38(2.30)	15.73(3.68)	$t(82) = -2.029$,	$p = 0.046^*$
diffPSCount	Spring	5.19(1.74)	4.85(2.06)	$t(72) = 0.765$,	$p = 0.446$
	Fall	7.54(1.57)	8.19(2.31)	$t(82) = -1.501$,	$p = 0.137$
WECount	Spring	7.85(1.17)	6.94(1.87)	$t(72) = 2.466$,	$p = 0.016^*$
	Fall	9.43(1.57)	9.52(2.37)	$t(82) = -0.210$,	$p = 0.833$
diffWECount	Spring	3.85(1.33)	2.61(1.87)	$t(72) = 3.226$,	$p = 0.002^*$
	Fall	2.15(1.42)	2.02(1.23)	$t(82) = 0.469$,	$p = 0.639$

Table 2: Log analysis results for the high competency groups per semester.

Feature	Semester	Random	DQN	Significance	
TotalCount	Spring	21.52(2.18)	24.31(4.07)	$t(35) = -2.471$,	$p = 0.021^*$
	Fall	24.27(0.76)	25.95(7.58)	$t(40) = -0.984$,	$p = 0.337$
PSCount	Spring	13.61(2.49)	17.50(4.67)	$t(35) = -3.008$,	$p = 0.006^*$
	Fall	14.59(0.66)	15.95(4.98)	$t(40) = -1.208$,	$p = 0.241$
diffPSCount	Spring	5.57(1.43)	5.12(2.30)	$t(35) = 0.680$,	$p = 0.502$
	Fall	7.68(0.83)	8.75(2.93)	$t(40) = -1.570$,	$p = 0.130$
WECount	Spring	7.90(1.33)	6.81(1.86)	$t(35) = 1.981$,	$p = 0.058\bullet$
	Fall	9.68(0.94)	10.00(3.19)	$t(40) = -0.428$,	$p = 0.672$
diffWECount	Spring	4.23(1.41)	2.43(1.82)	$t(35) = 3.271$,	$p = 0.002^*$
	Fall	2.18(1.46)	2.50(1.27)	$t(40) = -0.750$,	$p = 0.457$

mediate rewards” from the delayed rewards seems reasonable and works pretty well here. Thus, offline DRL can be successfully applied to real-life environments even with a limited training dataset with delayed rewards.

Acknowledgements

This research was supported by the NSF Grants #1432156, #1651909, and #1726550.

9. REFERENCES

- [1] M. Andrychowicz, B. Baker, et al. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*, 2018.
- [2] H. Azizsoltani and E. Sadeghi. Adaptive sequential strategy for risk estimation of engineering systems using gaussian process regression active learning. *Engineering Applications of Artificial Intelligence*, 74:146–165, 2018.
- [3] J. Beck, B. P. Woolf, and C. R. Beal. Advisor: A machine learning architecture for intelligent tutor construction. *AAAI/IAAI*, 2000(552-557):1–2, 2000.
- [4] M. Chi and K. VanLehn. Meta-cognitive strategy instruction in intelligent tutoring systems: How, when, and why. *Journal of Educational Technology & Society*, 13(1):25–39, 2010.
- [5] M. Chi, K. VanLehn, D. Litman, and P. Jordan. Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *UMUAI*, 21(1-2):137–180, 2011.
- [6] L. Cronbach and R. Snow. *Aptitudes and instructional methods: A handbook for research on interactions*. Oxford, England: Irvington, 1977.
- [7] S. D’Mello, B. Lehman, et al. A time for emoting: When affect-sensitivity is and isn’t effective at promoting deep learning. In *ITS*, pages 245–254. Springer, 2010.
- [8] J. T. Flam, S. Chatterjee, et al. On mmse estimation: A linear model under gaussian mixture statistics. *IEEE Transactions on Signal Processing*, 60(7):3840–3845, 2012.
- [9] S. Fujimoto, H. van Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.
- [10] D. Guo, S. Shamai, and S. Verdú. Mutual information and minimum mean-square error in gaussian channels. *arXiv preprint cs/0412108*, 2004.
- [11] T. Haarnoja, A. Zhou, et al. Soft actor-critic algorithms and applications. *arXiv:1812.05905*, 2018.
- [12] H. V. Hasselt. Double q-learning. In *Advances in Neural Information Processing Systems*, pages 2613–2621, 2010.
- [13] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [14] D. Horgan, J. Quan, et al. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018.
- [15] A. Iglesias, P. Martínez, R. Aler, and F. Fernández. Learning teaching strategies in an adaptive and intelligent educational system through reinforcement learning. *Applied Intelligence*, 31(1):89–106, 2009.
- [16] A. Iglesias, P. Martínez, R. Aler, and F. Fernández. Reinforcement learning of pedagogical policies in adaptive and intelligent educational systems. *Knowledge-Based Systems*, 22(4):266–270, 2009.
- [17] S. Kalyuga, P. Ayres, P. Chandler, and J. Sweller. The

- expertise reversal effect. *Educational psychologist*, 38(1):23–31, 2003.
- [18] N. Kim, Y. Lee, and H. Park. Performance analysis of mimo system with linear mmse receiver. *IEEE Transactions on Wireless Communications*, 7(11), 2008.
 - [19] T. P. Lillicrap, J. J. Hunt, et al. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
 - [20] T. Mandel, Y.-E. Liu, et al. Offline policy evaluation across representations with applications to educational games. In *AAMAS*, pages 1077–1084, 2014.
 - [21] B. M. McLaren and S. Isotani. When is it best to learn with all worked examples? In *AIED*, pages 222–229. Springer, 2011.
 - [22] B. M. McLaren, S.-J. Lim, and K. R. Koedinger. When and how often should worked examples be given to students? new results and a summary of the current state of research. In *CogSci*, pages 2176–2181, 2008.
 - [23] B. M. McLaren, T. van Gog, et al. Exploring the assistance dilemma: Comparing instructional support in examples and problems. In *Intelligent Tutoring Systems*, pages 354–361. Springer, 2014.
 - [24] V. Mnih, A. P. Badia, et al. Asynchronous methods for deep reinforcement learning. In *ICML*, pages 1928–1937, 2016.
 - [25] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.
 - [26] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.
 - [27] A. S. Najar, A. Mitrovic, and B. M. McLaren. Adaptive support versus alternating worked examples and tutored problems: Which leads to better learning? In *UMAP*, pages 171–182. Springer, 2014.
 - [28] K. Narasimhan, T. Kulkarni, and R. Barzilay. Language understanding for text-based games using deep reinforcement learning. *arXiv preprint arXiv:1506.08941*, 2015.
 - [29] X. B. Peng, P. Abbeel, et al. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)*, 37(4):143, 2018.
 - [30] X. B. Peng, G. Berseth, et al. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 36(4):41, 2017.
 - [31] D. Precup, R. S. Sutton, and S. P. Singh. Eligibility traces for off-policy policy evaluation. In *ICML*, pages 759–766. Citeseer, 2000.
 - [32] A. N. Rafferty, E. Brunskill, et al. Faster teaching via pomdp planning. *Cognitive science*, 40(6):1290–1332, 2016.
 - [33] A. Rajeswaran, V. Kumar, et al. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.
 - [34] C. E. Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.
 - [35] A. Renkl, R. K. Atkinson, et al. From example study to problem solving: Smooth transitions help learning. *The Journal of Experimental Education*, 70(4):293–315, 2002.
 - [36] R. J. Salden, V. Aleven, et al. The expertise reversal effect and worked examples in tutored problem solving. *Instructional Science*, 38(3):289–307, 2010.
 - [37] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.
 - [38] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In *ICML*, volume 37, pages 1889–1897, 2015.
 - [39] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
 - [40] D. Schwab and S. Ray. Offline reinforcement learning with task hierarchies. *Machine Learning*, 106(9-10):1569–1598, 2017.
 - [41] R. Schwonke, A. Renkl, et al. The worked-example effect: Not an artefact of lousy control conditions. *Computers in Human Behavior*, 25(2):258–266, 2009.
 - [42] S. Shen, M. S. Ausin, B. Mostafavi, and M. Chi. Improving learning & reducing time: A constrained action-based reinforcement learning approach. In *UMAP*, pages 43–51. ACM, 2018.
 - [43] S. Shen and M. Chi. Reinforcement learning: the sooner the better, or the later the better? In *UMAP*, pages 37–44. ACM, 2016.
 - [44] D. Silver, A. Huang, C. J. Maddison, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.
 - [45] D. Silver, T. Hubert, J. Schrittwieser, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.
 - [46] D. Silver, J. Schrittwieser, K. Simonyan, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.
 - [47] R. E. Snow. Aptitude-treatment interaction as a framework for research on individual differences in psychotherapy. *Journal of Consulting and Clinical Psychology*, 59(2):205–216, 1991.
 - [48] R. S. Sutton, D. A. McAllester, et al. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.
 - [49] J. Sweller and G. A. Cooper. The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction*, 2(1):59–89, 1985.
 - [50] T. Van Gog, L. Kester, and F. Paas. Effects of worked examples, example-problem, and problem-example pairs on novices’ learning. *Contemporary Educational Psychology*, 36(3):212–218, 2011.
 - [51] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, page 5. Phoenix, AZ, 2016.
 - [52] P. Wang, J. Rowe, W. Min, B. Mott, and J. Lester. Interactive narrative personalization with deep reinforcement learning. In *IJCAI*, 2017.

Optimizing Assignment of Students to Courses based on Learning Activity Analytics

Atsushi Shimada
Department of Advanced
Information Technology
Kyushu University, Japan
atsushi@ait.kyushu-
u.ac.jp

Hiroaki Ogata
Academic Center for
Computing and Media Studies
Kyoto University, Japan
hiroaki.ogata@gmail.com

Kousuke Mouri
Institute of Engineering
Tokyo University of Agriculture
and Technology, Japan
mourikousuke@gmail.com

Rin-ichiro Taniguchi
Department of Advanced
Information Technology
Kyushu University, Japan
rin@ait.kyushu-u.ac.jp

Yuta Taniguchi
Department of Advanced
Information Technology
Kyushu University, Japan
taniguchi@ait.kyushu-
u.ac.jp

Shin'ichi Konomi
Faculty of Arts and Science
Kyushu University, Japan
konomi@artsci.kyushu-
u.ac.jp

ABSTRACT

In this paper, we focus on optimizing the assignment of students to courses. The target courses are conducted by different teachers using the same syllabus, course design, and lecture materials. More than 1,300 students are mechanically assigned to one of ten courses taught by different teachers. Therefore, mismatches often occur between students' learning behavior patterns and teachers' approach to teaching. As a result, students may be less satisfied, have a lower level of understanding of the material, and achieve less. To solve these problems, we propose a strategy to optimize the assignment of students to courses based on learning activity analytics. The contributions of this study are 1) clarifying the relationship between learning behavior pattern and teaching based on learning activity analytics using large-scale educational data, 2) optimizing the assignment of students to courses based on learning behavior pattern analytics, and 3) demonstrating the effectiveness of assignment optimization via simulation experiments.

Keywords

Student assignment to courses, optimization, learning activity analytics

1. INTRODUCTION

Due to the widespread use of digital learning environments in education, collecting large-scale educational data has become easier in recent years. For example, online course educational systems such as Massive Open Online Courses (MOOCs) generate clickstream data from users who access the course websites. E-Learning systems such as Black-

board [5] and Moodle [9] record clickstream data when users submit reports, access materials, complete quizzes, etc. Educational data can also be extracted from e-Book systems (digital textbook systems), which provide precise logs of actions such as page movement, bookmarks, highlights, text memos, and so on. These large-scale educational data play a crucial role in the research domains of learning analytics and educational data mining.

Learning analytics is defined as the measurement, collection, analysis, and reporting of data about learners and their contexts for understanding and optimizing learning and the environments in which it occurs [1]. Various studies thus far have focused on learning analytics, including learning activity analysis [25], identifying at-risk students [17, 21], understanding learning paths [7], pattern mining [15], performance prediction [6, 14], and learning support [20].

In this paper, we focus on optimizing the assignment of students to courses. The optimization of assignment is often discussed for the purpose of timetabling problem [2, 18], teacher assignment to courses [8, 16], student assignment to courses [12, 19], and so on. The objective is to reduce the time consuming cost of educational office persons and faculty members, or to maximize the satisfaction of students and teachers. For these reasons, assignment problem is often applied to multi-different courses with consideration of the classroom capacities and preference of students and teachers. In contrast to these existing studies, the target courses of our study are conducted by different teachers using the same syllabus, course design, and lecture materials. More than 1,300 students are mechanically assigned to one of ten courses taught by different teachers. Therefore, mismatches often occur between students' learning behavior patterns and teachers' approach to teaching. As a result, students may be less satisfied, have a lower level of understanding of the material, and achieve less. To solve these problems, we propose a strategy to optimize the assignment of students to courses based on learning activity analytics.

The research questions and contributions of this study are

Atsushi Shimada, Kousuke Mouri, Yuta Taniguchi, Hiroaki Ogata, Rin-ichiro Taniguchi and Shinichi Konomi "Optimizing Assignment of Students to Courses based on Learning Activity Analytics" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 178 - 187

summarized as follows.

Research questions:

- RQ1.** Are learning activities common among courses or characterized by each individual course?
- RQ2.** Does better matching between the learning behavior pattern and teaching improve students' performance?

Contributions::

- C1.** Clarify the relationship between learning behavior pattern and teaching based on learning activity analytics using large-scale educational data.
- C2.** Optimize the assignment of students to courses based on learning behavior pattern analytics.
- C3.** Demonstrate the effectiveness of assignment optimization via simulation experiments.

In this paper, we review related research in the section 2 and then provide an overview of the proposed method including information about courses and the dataset in the section 3. The section 4 and section 5 discuss in detail the proposed method and strategy, and are followed by the discussion and conclusion in the section 6.

2. RELATED WORK

Optimization of assignment problem has been applied to several applications; such as timetabling problem [2, 18], classroom allocation problem [22], teacher assignment to courses [8, 16], student assignment to courses [24, 4, 12, 19], student grouping problem [11].

Elloumi et al. [2] defined the exam timetabling problem as the scheduling of exams to time slots, and the assignment of a set of exams to available classrooms. The objective was addressed to minimize the total capacity of the assigned classrooms. Phillips et al. [18] tackled the classroom assignment problem of university course timetabling. They solved an exact integer programming model for room assignment to get a Pareto optimal solution with respect to several solution quality measures on data from the university. Thongsanit [22] solved the classroom allocation problem. The number of students, the period of each course, the capacity of each classroom were used for optimization. Excel premium solver was applied to solve the problem.

Domenech et al. [8] solved the problem of teacher assignment to courses, taking teachers' preference into consideration. They developed a mixed integer linear programming model to balance teachers' teaching load and to maximize teachers' preference for courses. Ongy [16] also dealt with the teacher assignment problem to specific sections of particular courses. The assignment was solved to maximize the matching between teachers' competency to a specific subject. A mathematical model of the assignment process was formulated using mixed-integer programming.

Varone et al. [24] tackled the problem of course scheduling and assignment of students. They addressed students' preference for each course, a minimum number of students required to open a course, a maximum number of students for each course. The problem was defined as a generalization of

the student project allocation problem, and was solved by an integer programming problem. Ivo et al. [12] dealt with the problem of assigning students to elective courses according to their preference. They presented an integer programming model that maximizes the total student satisfaction in line with a number of different constraints. Shannon et al. [19] proposed an evolutionary algorithm for assigning students to courses. They addressed a situation where each student specified a set of courses with preference, and capacity of each course was given. The object was to maximize the overall student satisfaction by assigning each student to a course as high on his/her preference as possible.

As introduced above, optimization problems are often defined as a family of integer programming problem. One of common criteria is the capacity information such as classroom size, the number of students required by each course. In addition, taking preference of students or teachers into consideration will improve the satisfaction of them. In contrast to these studies, our study focuses on compulsory courses which all students have to join. The courses are conducted by several teachers in parallel, because of the limited capacity of each classroom. In compulsory courses, considering preference of students does not make much sense. Therefore, our method introduces a matching between learning behavior pattern and teaching which are objectively observed through the analytics of learning logs, instead of using subjective preference of students. To the best of our knowledge, our study is the first case to introduce the learning activity analytics results to optimizing student assignment to courses.

3. OVERVIEW OF METHODS

3.1 Lecture Course and Dataset

The dataset used in this study was collected from e-Learning and e-Book systems. The target courses were a series of lectures that constitutes the "Primary Course of Cyber Security," which commenced in Kyushu University in April 2018. Overall, 1,354 students were assigned to one of the 10 courses in advance. The lectures were conducted by six teachers in face-to-face style over seven weeks. Teachers followed the same syllabus and used the same lecture materials in the courses. Table 1 provides detailed information on the courses: teacher, course id and number of students. Note that in each course, four teachers were assigned to give two lectures each.

Table 1: Course Information

teacher	course id	students
Te01	60ab104927	114
Te01	6b1900c56c	120
Te02	9a683161f5	171
Te02	86066cba6d	143
Te03	792efa2c1b	139
Te03	34451e8c77	129
Te04	24a65f29b6	137
Te04	dbed6c966a	140
Te05	39a67f80f4	133
Te06	65bb6224af	128

All students have their own laptops and bring them to ac-

cess the e-Learning and e-Book systems during the lecture. We collected the learning activity logs over seven weeks. When an e-Book is operated, its timestamp, user id, material id, page number, and operation name are automatically recorded as an operation event. There are many types of operations; for example, OPEN indicates that a student has opened the e-Book file and NEXT indicates that the student has clicked the next button to move to the subsequent page. Students can bookmark a specific page, highlight selected characters, and make notes on a page. These operations correspond to the events ADD BOOKMARK, ADD MARKER, and ADD MEMO, respectively. A total of 4,087,730 e-Book operation logs were collected.

3.2 Analytics Flow

The analytics flow of this study comprises two stages. The first stage involves extracting the analytics of learning activities and quiz scores from each course. Statistical summaries of e-book operations, the browsing time for each page, and the distribution of the quiz scores for each lecture are analyzed to gather the characteristics of the courses. We then perform further detailed analytics of learning activities over courses to investigate the relationship between learning behavior patterns and quiz scores. We will show the possibility of optimizing the assignment of students to courses based on the results of these analytics.

At the second stage, we tackle the optimization issue, aiming to match learning behavior patterns and teaching to improve students' understanding of course contents. To this end, we use students' quiz scores instead of their level of understanding of course contents. We solve the optimization problem as a generalized assignment problem. We define a new cost function to realize the best assignment. We investigate the effectiveness of our assignment of students through simulation experiments.

4. LEARNING BEHAVIOR PATTERN ANALYTICS

There are several existing approaches to analyze learning behavior patterns in Massive Open Online Courses (MOOCs)[13, 10, 3]. On the other hand, our study focuses on learning logs collected during in-class, i.e., face-to-face lecture time, and out-class activities. Therefore, we newly design a methodology to analyze learning activities of students.

4.1 Course Activity Summary

The learning logs consist of four types of datasets: e-book operation logs, lecture material information, lecture time information, and quiz scores. First, we divide the e-book operation logs into in-class activity logs and out-class activity logs by referring to the lecture time information. With ten courses in the dataset, we acquire ten sets of in-class and out-class activity logs after the division procedure. Second, the in-class and out-class activity logs are aggregated page by page. The aggregation procedure is performed for each week (for seven weeks). This allows us to analyze the page-wise activity of each week. In addition, we calculate students' browsing time on each page by subtracting the timestamps between successive page transition events. Consequently, we acquire the total length of students' browsing time for each page.

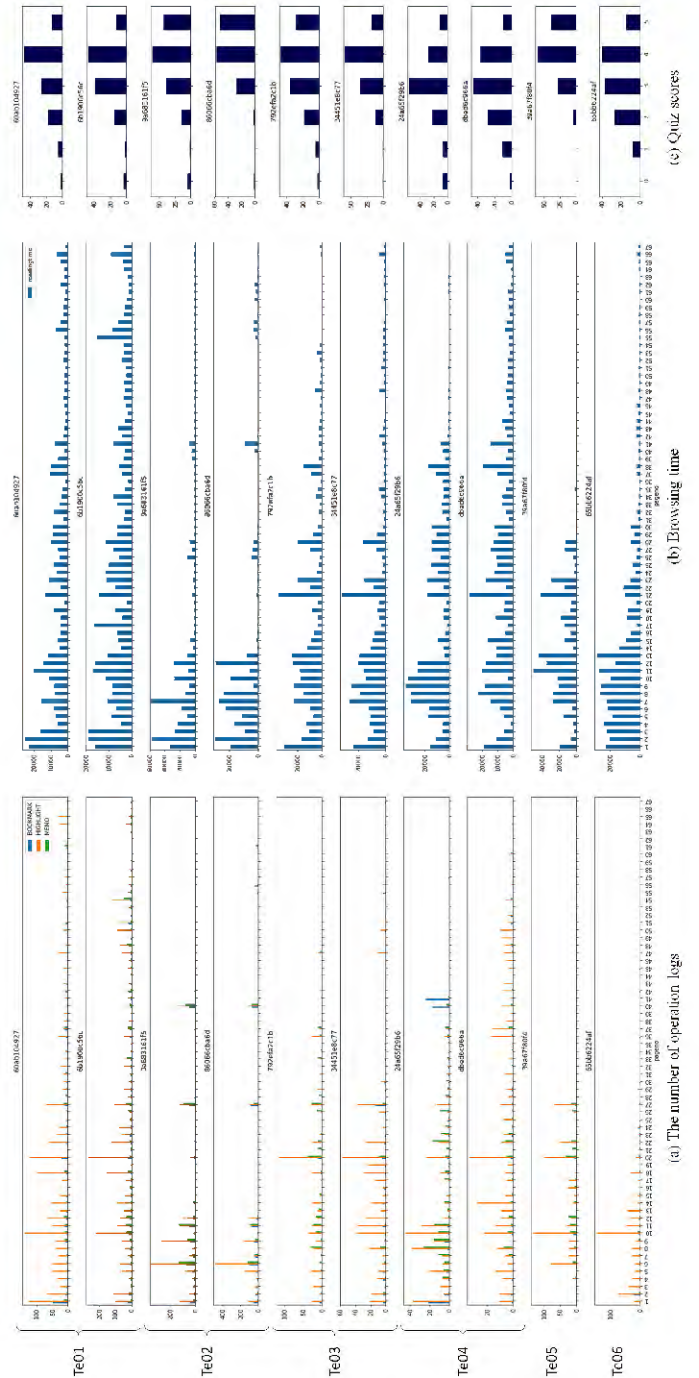


Figure 1: In-class Learning activity, browsing time and quiz scores of each course in the 1st week.

Figure 1 shows the visualization result of in-class activity, browsing time, and quiz scores for the first week (The figure is arranged with 90-degree rotation due to the page space limitation). The figure on the left displays page-wise e-book operations including “BOOKMARK,” “HIGHLIGHT,” and “MEMO.” The horizontal axis represents the page number while the vertical axis shows the number of operations aggregated by the students. Each row corresponds to a single course. The central figure shows students’ page-wise browsing time during lectures. The vertical axis of this figure is the timed duration (seconds). The figure on the right displays the quiz score distribution. After the lecture every week, students answered quizzes (averagely 5 questions). The quiz scores are normalized between 0 and 5 (full marks). The horizontal axis shows the scores and the vertical axis represents the number of students. The distributions of operations, browsing time, and quiz score are characterized for each course. For instance, the e-book operations are recorded in the former pages much more than latter pages. Regarding students’ browsing time, a longer time was spent on the former pages rather than latter pages. The quiz scores are also characterized by courses. The courses in the seventh and eighth rows received lower scores compared with other courses.

Next, let us focus on sets of two specific courses conducted by the same teachers. Of six teachers, four (Te01, Te02, Te03, and Te04) have two courses, as summarized in Table 1. We can see that the visualized results are similar for courses conducted by Te01, Te02, and Te03 compared with those of other teachers. Especially in the case of Te01 and Te02, the frequency of the e-book operation logs and browsing time for e-books have common peaks. On the other hand, in the case of Te04, the distributions are not so similar between two courses compared with the cases of other teachers. Even so, the similarity of the two distributions are higher than the courses conducted by the other teachers. Figure 2 shows the summary of e-book operation usage and quiz scores of each course in the first week. In the case of bookmark, highlight and memo operations, the value represents the average usage of each operation per page. The quiz score is normalized between 0 and 1. The higher value implies that students used the operations frequently or received better quiz scores. This figure illustrates that the courses conducted by the same teachers have similar values. While we show the result of the first week only due to the page space limitation, a similar tendency was observed in the other weeks.

From the above results, we inferred the following points. First, teachers have their own teaching ways, which do not differ widely between courses. Second, students’ learning activities are strongly affected by the teaching ways. To investigate these hypotheses, we further analyzed course characteristics.

4.2 Learning Activity Features

If learning activities are affected by teachers, the activities in each course should form a cluster, and the clusters related to the same teacher should have more similar features than the other clusters. For the investigation, we define a feature vector $F_{u,l}$ that represents the learning activities of student u for a lecture material l .

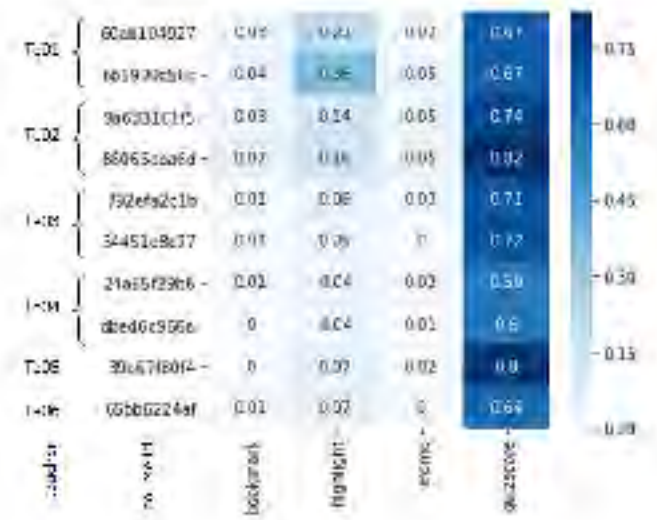


Figure 2: Learning activities in the 1st week.

To simplify the mathematical formulation, the notation u is omitted from the following explanation. Let f_p be a page-wise feature vector in page p of the lecture material. The f_p has eight elements;

$$f_p = (b_p^i, h_p^i, m_p^i, t_p^i, b_p^o, h_p^o, m_p^o, t_p^o), \quad (1)$$

where b_p^* , h_p^* , and m_p^* are the number of operation logs of “BOOKMARK,” “HIGHLIGHT,” and “MEMO” recorded during $(*=i)$ /outside $(*=o)$ the lecture time. The t_p^i and t_p^o are the browsing time of page i during the lecture time and outside lecture time, respectively. A feature vector for a specific lecture material l containing l_N pages is defined by the concatenation of f_p as

$$F_l = (f_1, \dots, f_p, \dots, f_{l_N}). \quad (2)$$

For instance, when a lecture material l consists of 50 pages, the feature vector has 400 (8-dim \times 50 pages) dimensions. Note that, in fact, the feature vector is calculated for each student u defined as $F_{u,l}$.

We apply t-SNE (t-Distributed Stochastic Neighbor Embedding) [23] to investigate the similarity and dissimilarity of feature vectors within the course and among the courses. t-SNE is a technique for dimensionality reduction. It is often used for the visualization of high-dimensional datasets. It converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of low-dimensional embedding and high-dimensional data. Figure 3 shows the visualization result in two-dimensional space. Courses are marked by color. We can see that the feature vectors distribute closely in the same course, while those of other courses make distinguishable clusters. From these results, we can say that learning activities are affected by teachers, as mentioned in the previous section.

4.3 Learning Activity vs. Quiz Score

Through analyzing e-book operation logs and learning activity features, we found that the learning activity itself is

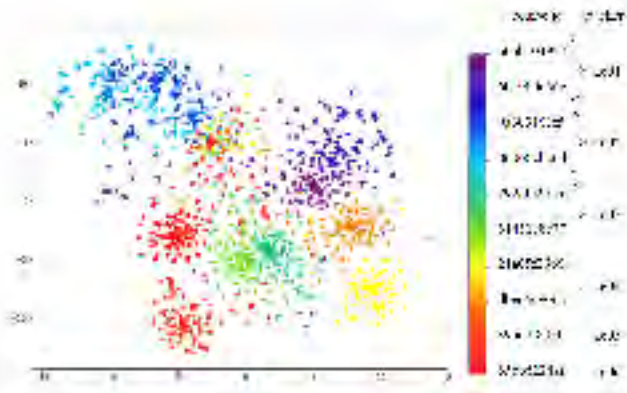


Figure 3: Visualization of feature vectors by t-SNE.

characterized by courses, i.e., teachers who conducted the lectures. On the other hand, the relationship between learning activities and quiz scores was not addressed in previous analytics. Although learning activity (i.e., feature vector $F_{u,l}$) is similar in the same course, quiz scores are distributed widely, as shown in the right part of Figure 1. Therefore, it is important to perform a relation analysis between learning activities and quiz scores.

To extract the characteristics of learning activities, we apply the k-means clustering method to the set of feature vectors $F_{u,l}$ acquired from all students. Then, we investigate the quiz score of each cluster and each course. Note that the clustering is performed for all feature vectors without considering the course id information. In other words, the learning activity features are purely analyzed to generate clusters. Afterwards, we put the course id again to each feature vector to investigate the clustering result. Figure 4 shows the result of the first week when the number of clusters was set to be 5. The horizontal axis is the course id and the vertical axis is the cluster id (from 0 to 4, totally 5 clusters). The value of each cell indicates the average quiz score. For example, in the left column and fifth row, the score is 3. This means that the students in course id “60ab104927,” with learning activity in cluster id “4” received the score of 3 on an average. The cells with values of zero indicate that no student belongs to the cluster or the course. The detailed distribution of quiz scores in each cluster is shown in Figure 5. The horizontal axis is the cluster id, and the vertical axis is the number of students over courses. We can see that each cluster cannot be explained by quiz scores. Even in the same cluster, that is, even in the similar learning activity, some students received better scores while others received worse scores.

Figure 4 displays interesting and important characteristics of lectures. First, some clusters (e.g., cluster id 0) represent the characteristics of learning activities observed only in limited courses (e.g., course id “9a683161f5” and “86066c6ba6d”). In the case of cluster id 1, the corresponding learning activities are observed in all courses, but the average quiz scores are different. Students in course id “86066c6ba6d” received higher scores, while those in course id “24a65f29b6” received lower scores than in the other courses. On the other hand,



Figure 4: Average quiz scores of each cluster and each course when the number of clusters was 5 in the 1st week.

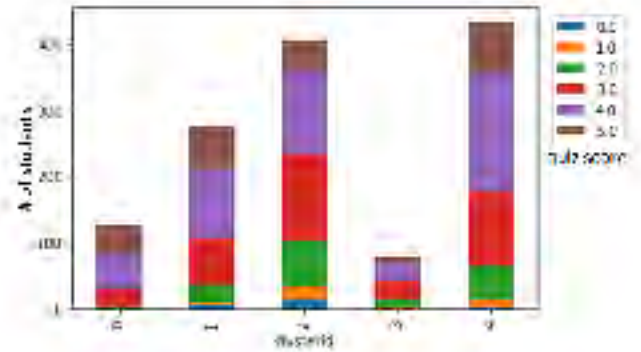


Figure 5: Quiz score description of each cluster in the 1st week.

each column also shows interesting characteristics of each course. For example, students belonging to cluster id 2 received high scores in course id “39a67f80f4,” while those in cluster id 3 received lower scores. In the case of course ids “60ab104927” and “792efa2c1b,” the scores of cluster id 3 are higher than those of cluster id 2. Therefore, our findings are summarized as follows. Even if learning activities are similar, quiz scores differ among courses. The characteristics of each course and its method of scoring quizzes are different for each cluster. We investigated these characteristics by changing the number of clusters from 3 to 29 (14 patterns) and found the same results. Due to length limitations, we only show additional results when the number of clusters was set to 15 in Figure 6. As the number of clusters increases, course-specific clusters appear, such as cluster id 1, 2, and 3.

5. OPTIMIZATION OF STUDENT ASSIGNMENT TO COURSES

Based on the results of the learning activity analysis and findings in the previous section, we optimize the process of assigning students to courses considering learning activities and course characteristics. In this section, we define the characteristic of courses c as the ability to give a student group g a quiz score $A_{c,g,l}$ on average for the lecture material l . Note that the lecture material l completely corresponds to each week, so that we can regard the l as the indicator of week. The c , g , and $A_{c,g,1}(l = 1)$ correspond

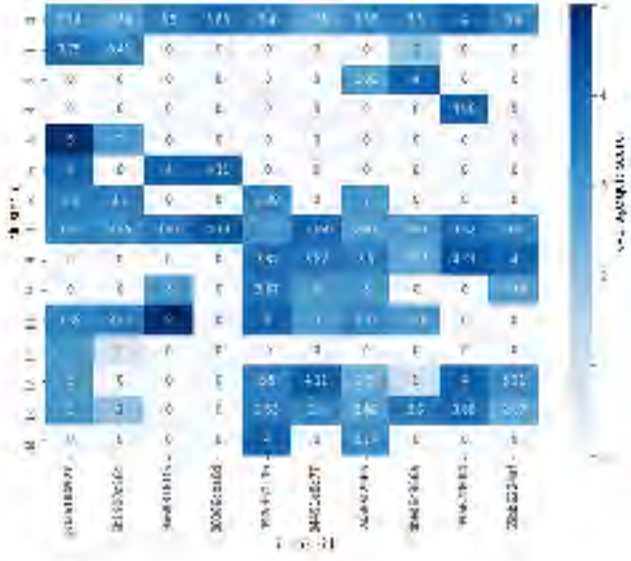


Figure 6: Average quiz scores of each cluster and each course when the number of clusters was 15 in the 1st week.

to the column, row, and element value in Figure 4, respectively. Our assumption is that students will get better quiz scores if they move to better-suited courses. For example, students in group $c = \text{"dbed6c966a"}$ and $g = 3$ received an average quiz score of 2.0 (i.e., $A_{c,g,1} = 2.0$). If they could move to another course, "792efa2c1b" , their quiz score would become 1.89 points higher (will receive 3.89 points on average). While this is an ideal situation, we suppose that a good match between course characteristics and learning behavior patterns will generate positive effects. Therefore, we propose an optimized strategy of assigning students to courses based on learning activity analytics.

5.1 Assignment Problem

The optimization of assignment can be considered the generalized assignment problem (GAP). The GAP is a problem in combinatorial optimization in which each agent in one set is matched to a single task in another set. Each task has a limited capacity for agents, and the goal is to minimize the sum of the costs or maximize the sum of profits. Formally, the problem can be stated as an integer programming problem.

In the case of our study, the agents and tasks can be replaced by the courses and students. The problem is:

$$\text{minimize } \sum_{c=1}^C \sum_{u=1}^U w_{c,u} x_{c,u} \quad (3)$$

$$\text{subject to } \sum_{u=1}^U x_{c,u} \geq S_c, \quad \text{for } c = 1, \dots, C \quad (4)$$

$$\sum_{c=1}^C x_{c,u} = 1, \quad \text{for } u = 1, \dots, U \quad (5)$$

where C is the number of courses, U is the number of students, and $w_{c,u}$ is the cost for the assignment of student u to course c . The detailed definition of $w_{c,u}$ will be explained later. The $x_{c,u}$ becomes 1 if student u is assigned to course c ; otherwise, it is zero. The S_c is the minimum of students required in course c .

We define the cost $w_{c,u}$ as follows:

$$w_{c,u} = (H - A_{c,m(u)}) + b \quad (6)$$

where H is the maximum quiz score, $m(u)$ is a map function that presents the group g (i.e., cluster id) to which the student u belongs, and b is the bias term to penalize changing courses. The first term $(H - A_{c,m(u)})$ becomes smaller when student u is assigned to a course c in which student u will likely receive a higher quiz score. In other words, student u belonging to group $m(u)$ is likely to move to a course that gives higher quiz scores for group $m(u)$. Note that $m(u)$ indicates the student group (cluster id) that has a similar learning activity within the group, so that we can estimate a quiz score $A_{c,m(u)}$ for every assignment situation because $A_{c,m(u)}$ corresponds to an element in Figure 4. The bias term b gives an additional cost to constrain the course movement (change of the assignment from one course to another). If we give a large value to b , the $w_{c,u}$ also becomes large, which most likely results in students remaining in the current course. Note that the above equations are calculated in each week so that the notation l , which identifies the lecture material, should be put on each term such as $w_{c,u,l}$, $x_{c,u,l}$, $A_{c,m(u),l}$. In the above equations, we omitted the notation l to simplify the mathematical formulation.

5.2 Assignment Results

We conducted experiments to investigate the proposed assignment strategy. We varied the number of clusters from 3 to 29 (14 patterns) and the bias b from 0.0 to 1.0. We performed k-means clustering to acquire the relation matrix of the quiz scores between student groups g and courses c as shown in Figure 4. Next, we solved the generalized assignment problem by changing the bias value to 0.0, 0.1, 0.3, 0.5, and 1.0. Assignment optimization was conducted for each week individually so that we acquired a total of 490 assignment results (14 clustering patterns \times 5 bias patterns \times 7 weeks). In the following paragraph, we report how the assignment result changed according to the number of clusters and the strength of the bias.

First, we investigated how many students were assigned (moved) to the other courses. Figure 7 shows the assignment results when the number of clusters was 5 and the bias was 0.5. Course ids are arranged in horizontal and vertical lines. The horizontal line indicates the course id to which students belonged before the assignment, that is, the original course assigned by the university. The vertical line shows the course id to which students were assigned after the optimization of the assignment problem. The value of each cell is the number of students. For example, 137 of 139 students who originally belonged to "86066cba6d" remained in the same course, but 2 students moved to the course "9a683161f5" . In the case of this result, the bias was set to be 0.5, which is a relatively strong bias to constrain course changes, resulting in students remaining in original courses (large value in diagonal element) rather than changing courses. Interestingly, a

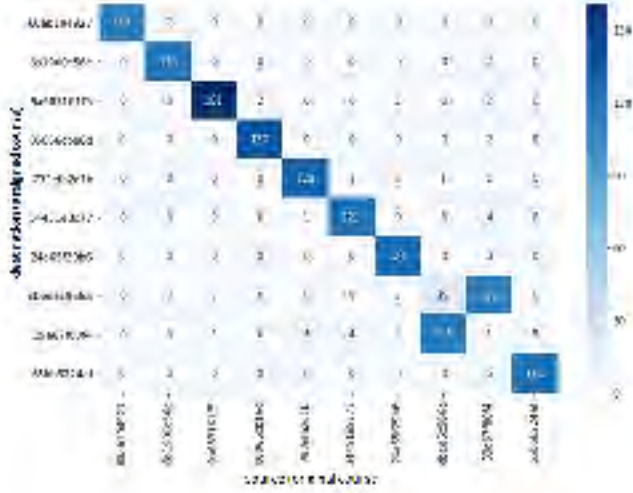


Figure 7: Movement matrix when the number of clusters was 5 in the first week.

large movement occurred between course “dbed6c966a” and “39a67f80f4.” Figure 8 shows another result when the number of clusters was 15 and the bias was 0.0 (no bias). Compared with Figure 7, a larger number of students moved from their original courses to other courses.

Next, we investigated the total number of students who moved courses. Figure 9 shows the summarized result of course movement. The horizontal axis shows the number of clusters that we set when performing k-means clustering for learning activity logs. The vertical axis displays the percentage of students who were assigned to the other courses. The five lines represent the results with different values of bias. In general, as the number of clusters increased and as the value of bias decreased, many students were assigned to other courses. The larger number of clusters generated small clusters that precisely indicate the representative learning activities, which is why the flexibility of the matching between students and courses increased. In the case of no bias (bias $b = 0.0$), students moved among courses the most flexibly.

The flexibility directly related to the encouragement of quiz scores. Figure 10 shows the improved quiz scores after the optimization of student assignments. The vertical axis is the value of the improved quiz score. In fact, the improvement in the quiz score $\hat{q}_{u,l}$ of each student u for the lecture material l (i.e., l th week) was calculated by:

$$\hat{q}_{u,l} = q_{u,l}^{org} - A_{c,\hat{m}(u),l} \quad (7)$$

where $q_{u,l}^{org}$ is the original quiz score of student u in l th week and $\hat{m}(u)$ is the map function that gives the group id (cluster id) to which student u was assigned. The line graphs indicate the average of score $\hat{q}_{u,l}$ of all students over seven weeks. We can see the similar tendency of the line graphs compared with Figure 9. We further investigated the improvement of quiz scores in each week. We identified three typical cases:

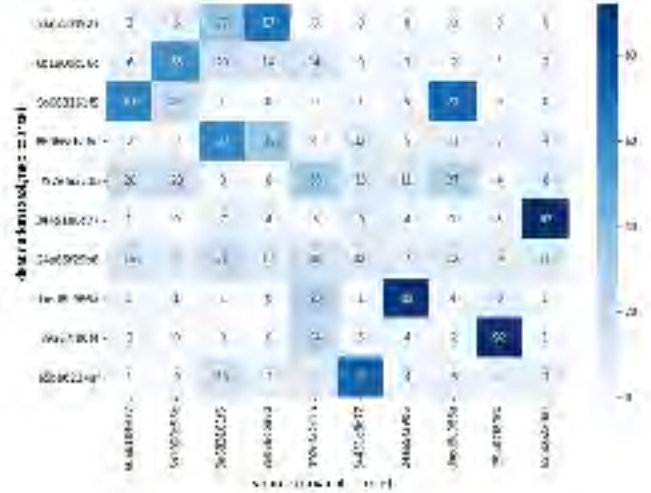


Figure 8: Movement matrix when the number of clusters was 15 in the first week.

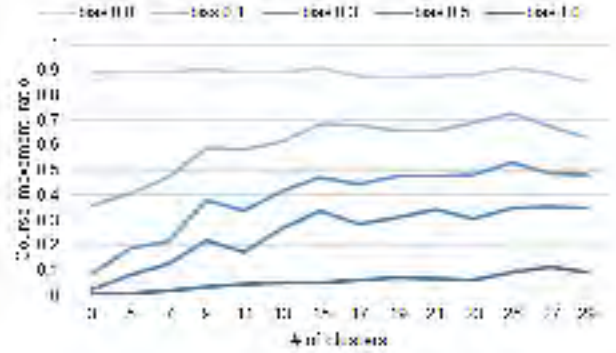


Figure 9: Course movement over 7 weeks.

2nd week: w2 the original score was higher on average than in other weeks.

4th week: w4 the original score was lower on average than in other weeks.

6th week: w6 the original score was at an average level over 7 weeks.

Figure 11 shows the line graphs of three cases; the solid line and dashed line correspond to the different settings of bias value at 0.0 and 0.5, respectively. The lower the original score (fourth week), the more the score was improved. From these results, we can expect to improve the quiz scores through the optimization of student assignments. The level of improvement is affected by the number of clusters and the value of bias. In addition, the number of course movements is strongly affected by the bias.

Finally, we note again that the experiments in this section demonstrate the success of the proposed optimization strategy based on the analytics of learning activities. Although

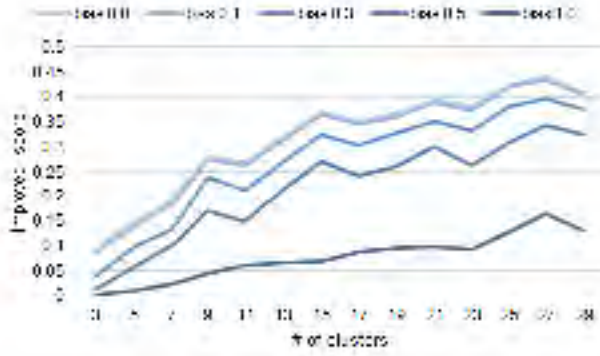


Figure 10: Score improvement after optimization of student assignment to courses.

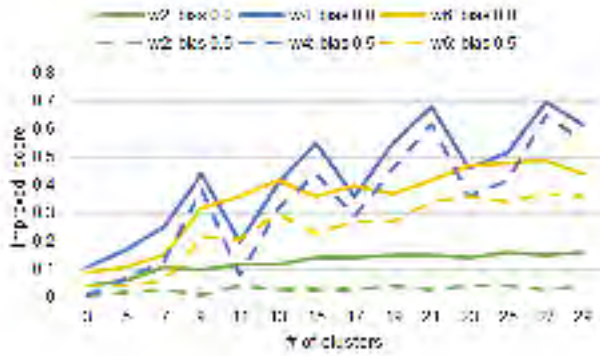


Figure 11: Three typical examples of score improvement.

the improvement of quiz score $\hat{q}_{u,l}$ is a simulated result, we can expect students to get better scores if matching between students and courses is optimized. This expectation comes from the fact that matching optimization provided better scores, as shown in Figure 4 and 6 of the previous section.

5.3 Simulation in a Realistic Situation

In this section, we conduct simulated experiments considering a more realistic situation. The proposed optimization method requires students' learning activity logs to analyze learning behavior patterns and quiz scores. Therefore, we assume that the optimization of student assignment is performed after the lecture of the first week. Using the data from learning activities and quiz scores collected in the first week, we optimize the assignment of students to courses. The assignment then remains the same after the second lecture. We simulated the quiz scores of students who were assigned to another course after the first lecture from the second to seventh week.

This simulation is difficult because although the actual quiz scores in the original courses are known, quiz scores after the optimization of assignments are unknown. Therefore, we must estimate students' quiz scores after optimization. We will now review the purpose of clustering learning activi-

ties. As concluded in the previous section, students received different quiz scores even when their learning activities were similar. After the optimization of student assignment to courses, students who received worse scores in the original course should be moved to another course in which they will receive better quiz scores. Therefore, we will focus on a student who has a similar learning activity in the course to which the target student is assigned. More specifically, let a target student be y and consider a situation where student y is assigned to course c . For all students who originally belonged to course c , we search for a student z who has the most similar learning activity to student y . Mathematically,

$$z = \arg \min_u |F_{y,l} - F_{u,l}| \quad (8)$$

where $F_{u,l}$ is a feature vector of learning activity of student u for the lecture material l . The lecture material l corresponds to a specific lecture, so that we can regard l as the lecture conducted in each week. Finally, we regard the original quiz score $q_{z,l}^{org}$ of student z as the estimated score of the target student y . Let $\hat{q}_{y,l}^{new}$ be the estimated quiz score of student y after the assignment. The score improvement ratio $r_{y,l}$ can be calculated by:

$$r_{y,l} = \frac{\hat{q}_{y,l}^{new} - q_{y,l}^{org}}{H - q_{y,l}^{org}} \quad (9)$$

where H is the maximum quiz score (the same with eq. 6) and q_y^{org} is the quiz score of student y in the original course. Note that $q_{y,l}^{org}$ is the actual quiz score and $\hat{q}_{y,l}^{new}$ is the estimated quiz score. Figure 12 illustrates the overview of the simulation strategy. In the Figure, student y is assigned to course A after the 1st week. The quiz scores from the 2nd week to 7th week have to be estimated because the student y has the quiz scores in the original course B. Our strategy explore the most similar (matched) feature vector $F_{u,l}$ from the students in course A. In the case of this figure, the learning activities of student 1 and student 2 are the best matched in the 2nd week and 3rd week, respectively. As the results, the quiz score of $q_{1,2}^{org}$ and $q_{2,3}^{org}$ are used for the estimated quiz scores of student y in the 2nd week and 3rd week, respectively.

Figure 13 shows the simulation result. The horizontal axis shows the number of clusters, and the vertical axis represents the score improvement ratio. The improvement ratio is averaged over six weeks (from the 2nd to 7th week) and over students who moved from the original course to the other course. Totally, the improvement ratio of every setting (any number of clusters, or any value of biases) was higher than zero, which means that the student assignments created positive effects for students. The scale of the effect was the largest when we set the bias b to 1.0. In contrast to the result in Figure 10, the largest value of bias provided the best result. This is because the improvement ratio is summarized by students who were assigned to the other course only. In the case of a large value of bias b , the movement from one course to another is constrained, so that a small number of students actually changed courses, as shown in Figure 9. As a result, the optimization of student assignment provided higher effects (i.e., made students receive better quiz scores) for a limited number of students. As the number of students increases (the bias decreases), the effect becomes smaller due to averaging calculation of improvement ratios. There was

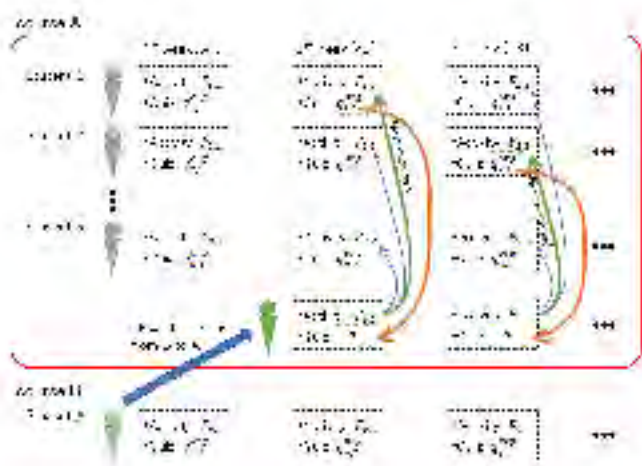


Figure 12: Overview of simulation experiment how to estimate the quiz scores in the newly assigned course

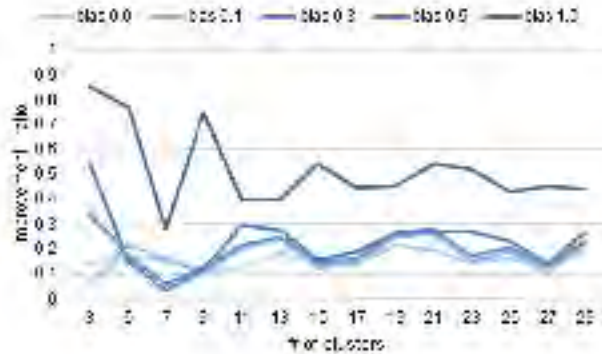


Figure 13: Average score improvement ratio over 6 weeks.

not a large difference of improvement ratios among the four settings where the bias value was 0.0, 0.1, 0.3, or 0.5 when the number of clusters was larger than five. In terms of the calculation cost and ease in explaining/interpreting learning activities, we prefer the smaller number of clusters, so that selecting the five clusters is one of the reasonable solutions.

Finally, Figure 14 shows the weekly improvement ratio when the number of clusters was fixed to be five. From the fourth (w4) to the seventh week (w7), a similar tendency was observed: The large value of bias b provided higher improvement. On the other hand, in the second (w2) and third weeks (w3), even the smaller value of bias b provided better results. We guess that the factor comes from the calculation of improvement ratio. When the original quiz scores are close to the maximum quiz score H , the denominator eq. 9 becomes smaller, resulting in a larger improvement ratio. In fact, the average of the original quiz scores in the second week was quite higher than in other weeks. In terms of maximization of the number of students who are supposed to get better quiz scores, the constraint bias should be relaxed as

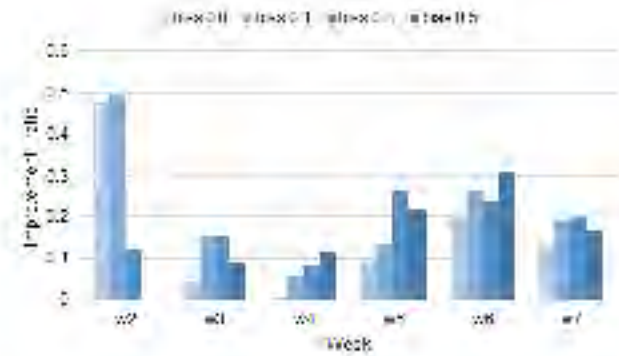


Figure 14: Average score improvement ratio of each week when the number of clusters was 5.

least as possible. Therefore, a reasonable guideline to set the number of clusters and bias is to set a smaller number of clusters (such as 5 or 7) and set a smaller value of bias (such as 0.1, 0.3, or 0.5).

6. DISCUSSION AND CONCLUSION

We proposed a strategy to optimize the assignment of students to courses based on learning activity analytics. This optimization is first intended to minimize the mismatch between students' learning behavior patterns and teachers, and second to maximize the improvement of students' quiz scores by assigning them to courses that are more suited to their learning behavior patterns. The success of these interventions are supported by the learning analytics results. Analyzing e-book operation logs and quiz scores collected from 1,354 students in 10 courses based on the same syllabus and lecture materials, we identified the following findings. From the macro perspective, learning activities are affected by teachers. Although teachers are not directly observed by teaching logs, the patterns implicitly appear as course-specific features, as shown in Figure 3. From the micro perspective, students' learning activities can be grouped into several clusters, each cluster representing a feature of such learning activities. Regardless of courses, students have similar learning activity features if they belong to the same cluster. On the other hand, quiz scores differ among students who belong to the same cluster. From these facts, we formulated the hypothesis that good matching between learning behavior pattern and teaching approach would provide better effects for students.

Our proposed approach requires learning activity logs to be acquired before students can be assigned to courses. Thus, in our experiments, we used learning logs collected in the first week and then optimized the assignment of students for subsequent weeks. Another promising solution to this matter could employ learning logs collected in other lecture courses or in past courses. If such logs are available, learning activities can be analyzed in advance and students can be optimally assigned to courses before the first week's lecture begins. This paper showed the effectiveness of an optimization strategy through the results of simulation experiments in which quiz scores improved. Meanwhile, the proposed method has another important aspect as a useful tool to

simulate the effects of assignments in advance. In future work, we will investigate the effectiveness of the optimization strategy in live settings based on effect simulations.

Acknowledgements

This work was supported by JST PRESTO Grant Number JPMJPR1505, JSPS KAKENHI Grant Number JP16H06304 and JP18H04125, Japan.

7. REFERENCES

- [1] <https://solaresearch.org/> (SoLAR).
- [2] ABDELKARIM, E., HICHEM, K., BASSEM, J., AND ABDELAZIZ, D. The classroom assignment problem: Complexity, size reduction and heuristics. *Appl. Soft Comput.* 14 (Jan. 2014), 677–686.
- [3] AN, T.-S., KRAUSS, C., AND MERCERON, A. Can typical behaviors identified in moocs be discovered in other courses? In *The Tenth International Conference on Educational data Mining (EDM 2017)* (2017), pp. 220–225.
- [4] BINYAMIN, K., JON, L., AND DANIEL, N. Optimizing the assignment of students to classes in an elementary school. *INFORMS Transactions on Education* (2014), 38–44.
- [5] BRADFORD, P., PORCIELLO, M., BALKON, N., AND BACKUS, D. The Blackboard Learning System: The Be All and End All in Educational Instruction? *Journal of Educational Technology Systems* 35, 3 (Apr. 2007), 301–314.
- [6] BRINTON, C. G., AND CHIANG, M. Mooc performance prediction via clickstream data and social learning networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)* (April 2015), pp. 2299–2307.
- [7] DAVIS, D., CHEN, G., HAUFF, C., AND HOUBEN, G. Gauging MOOC learners’ adherence to the designed learning path. In *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016* (2016), pp. 54–61.
- [8] DOMENECH, B., AND LUSA, A. A milp model for the teacher assignment problem considering teachers’ preferences. *European journal of operational research* 249, 3 (Jan 2016), 1153–1160.
- [9] DOUGIAMAS, M., AND TAYLOR, P. Moodle: Using learning communities to create an open source course management system. In *Proceedings of EdMedia: World Conference on Educational Media and Technology 2003* (Honolulu, Hawaii, USA, 2003), D. Lassner and C. McNaught, Eds., Association for the Advancement of Computing in Education (AACE), pp. 171–178.
- [10] FERGUSON, R., AND CLOW, D. Consistent commitment: Patterns of engagement across time in massive open online courses (moocs). *Journal of Learning Analytics* 2 (01 2015), 55–80.
- [11] HÜBSCHER, R. Assigning students to groups using general and context-specific criteria. *IEEE Transactions on Learning Technologies* 3, 3 (2010), 178–189.
- [12] IVO, B., AND JOSKO, M. An integer programming model for assigning students to elective courses. *Croatian Operational Research Review* 6 (2015), 511–524.
- [13] KIZILCEC, R. F., PIECH, C., AND SCHNEIDER, E. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *LAK* (2013), ACM, pp. 170–179.
- [14] MOURI, K., OKUBO, F., SHIMADA, A., AND OGATA, H. Bayesian network for predicting students’ final grade using e-book logs in university education. In *IEEE International Conference on Advanced Learning Technologies (ICALT2016)* (2016), pp. 85–89.
- [15] OI, M., OKUBO, F., SHIMADA, A., YIN, C., AND OGATA, H. Analysis of preview and review patterns in undergraduates’ e-book logs. In *The 23rd International Conference on Computers in Education (ICCE2015)* (2015), pp. 166–171.
- [16] ONGY, E. E. Optimizing student learning: A faculty-course assignment problem using linear programming. *Journal of Educational and Human Resource Development* 5 (2017), 1–14.
- [17] PARK, J., DENARO, K., RODRIGUEZ, F., SMYTH, P., AND WARSCHAUER, M. Detecting changes in student behavior from clickstream data. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference* (2017), pp. 21–30.
- [18] PHILLIPS, A. E., WATERER, H., EHRGOTT, M., AND RYAN, D. M. Integer programming methods for large-scale practical classroom assignment problems. *Computers & OR* 53 (2015), 42–53.
- [19] SHANNON, C. A., AND MCKINNEY, D. An evolutionary algorithm for assigning students to courses. In *Proceedings of the Twenty-Fourth International Florida Artificial Intelligence Research Society Conference* (2011), pp. 388–393.
- [20] SHIMADA, A., OKUBO, F., YIN, C., AND OGATA, H. Automatic summarization of lecture slides for enhanced student preview-technical report and user study-. *IEEE Transactions on Learning Technologies* 11, 2 (2018), 165–178.
- [21] SHIMADA, A., TANIGUCHI, Y., OKUBO, F., KONOMI, S., AND OGATA, H. Online change detection for monitoring individual student behavior via clickstream data on e-book system. In *8th International Conference on Learning Analytics & Knowledge* (3 2018), pp. 446–450.
- [22] THONGSANIT, K. Solving the course - classroom assignment problem for a university. *Silpakorn University Science & Technologies Journal* 8, 1 (2014), 46–52.
- [23] VAN DER MAATEN, L., AND HINTON, G. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.
- [24] VARONE, S., AND SCHINDL, D. Course opening, assignment and timetabling with student preferences. In *Proceedings of the 2nd International Conference on Operations Research and Enterprise Systems* (2013).
- [25] WANG, G., ZHANG, X., TANG, S., ZHENG, H., AND ZHAO, B. Y. Unsupervised clickstream clustering for user behavior analysis. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (2016), CHI ’16, pp. 225–236.

Towards the Prediction of Semantic Complexity Based on Concept Graphs

Rémi Venant
Le Mans University
LIUM - EA 4023, Le Mans Université
72085 Le Mans, Cedex 9, France
remi.venant@univ-lemans.fr

Mathieu d'Aquin
Insight Centre for Data Analytics
National University of Ireland
Galway
mathieu.daquin@insight-centre.org

ABSTRACT

The evaluation of text complexity is an important topic in education. While this objective has been addressed by approaches using lexical and syntactic analysis for decades, semantic complexity is less common, and the recent research works that tackle this question rely on machine learning algorithms that are hardly explainable and are not specifically designed to measure this variable. To address this issue, we explore in this paper the engineering of novel features to evaluate conceptual complexity. Through the construction of a knowledge graph that captures the concepts present in a text and their generalized forms, we measure different graph-based metrics to express such a complexity. Eventually, early-stage evaluations based on a well-known public corpus of students' productions show that the use of these metrics significantly improves performance compared to a state-of-the-art binary neural network classifier.

Keywords

semantic complexity, concept complexity, knowledge graph, features engineering, neural network, machine learning

1. INTRODUCTION

In Technology Enhanced Learning, the evaluation of the complexity of textual material underlies several activities. For instance, assessments in language classes are based, among other things, on the measurement of the learners' abilities to deal with different grammatical structures or to use the most precise vocabulary to express their thoughts. Another example can be found in learning resources indexation. Merlo [4] and Openstax¹ are two projects that aim at offering fine-grain information retrieval functionalities and automatic recommendation systems for educational objects based on their metadata, including their level of difficulty.

Complexity is usually related to readability, a concept defined as the "total sum of all those elements within a given

¹<https://openstax.org>

piece of printed material that affect the success a group of readers have with it" [5]. Thus, readability depends on both the object (the text) and the subject (the reader), whereas complexity is commonly characterized by a function whose output does not differ from one reader to another [11, 21].

Many research works in Natural Language Processing (NLP) sought out a way to measure the complexity of a text, and to predict the category (e.g. the level of difficulty) it should fall into. Most of them, nevertheless, focus on lexical and syntactic evaluation to achieve their objectives. Alongside the recent progress in machine learning, and more specifically in deep learning, another approach to text classification arose, based on semantic relationship of words within a text [17]. While offering outstanding performance, these predictive models are not easily explainable [28]. However, to provide this property is of importance to increase the confidence a user gives to these systems [1], and would allow to improve the usability of predictive models. For instance, a tutoring system designed for learning a foreign language would benefit from such a model, that is able not only to assess the complexity of a student's writing, but also to give suggestions on how to improve it.

Hence we propose in this paper a novel approach to measure semantic complexity with a model based on explainable features: we exploit the semantic web to build the conceptual representation of a text within an ontological graph, that we use to compute a set of metrics. In order to validate our model, we focus here on the following research questions:

- Is a model based on our sole metrics able to outperform a state-of-the-art model of semantic complexity?
- Does the extension of a state-of-the-art predictive model with our engineered features improves its performance?

In this paper, we answer these questions in the context of complexity assessment of students' productions in English as a foreign language. The first section is dedicated to related work in complexity assessment, in order to select a state-of-the-art model to compare to. We describe then the pipeline we designed to build a conceptual graph from a text and convert it into a vector, before going into the details of the 17 metrics that compose the vector. The fourth section is dedicated to the analysis of our model, in order to answer the research questions we defined previously.

Rémi Venant and Mathieu d'Aquin "Towards the prediction of semantic complexity based on concept graphs" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 188 - 197

2. RELATED WORK

Complexity, as a function of the text only [29], is involved in different tasks related to education. Classification of resource materials has become an important topic with the growth of open education and MOOCs [15], in order to provide suitable recommendations to a learner [3]. Learner evaluation would also benefit from such a function. Checking mistakes is not enough to assess the writing skill of a student; one also needs to measure the ability to express more complex thoughts, with the use of precise concepts [19]. A last instance of a task that requires complexity assessment is text simplification, a process that aims at providing a simpler version of a text by reducing its complexity, without removing its substantial content [25].

These objectives require computational models in order to classify them or to measure a value of complexity according to the needs. Different categories of metrics are involved to build such models, and most of the current research works rely on surface, syntactic and lexical features [8, 7, 24]. Surface level features provide basic statistical measures such as the number of characters or words. Lexical features target the structure of sentences, such as the average number of verb phrases per sentences or the number of dependent clauses [21]. Finally, syntactic features are based on the recognition of terms to compute metrics such as the average number of synonyms of words. Other morphological values (i.e., based on the structure of text) can be found. Siddharthan [29] proposed to measure discourse complexity to evaluate whether connections between text segments are vague or weak. In addition, Davoodi [8] used coherence features, that refer to the grammatical and lexical links which connect linguistic entities together, in order to include the influence of discourse structure on text complexity assessment. Similar features can be found in [24], who evaluated the lexical cohesion and the level of argumentation.

With these features, predictive models of complexity have already shown good performances. For instance, a SVM binary classifier using 117 parameters that belong to these categories of features achieved to classify Swedish texts according to their complexity with an accuracy of 98.9% [12]. Although widely used, surface and syntactic features work only on the structure of the text, while lexical features, using a base of knowledge for word recognition do not provide any semantic to these words. Thus, a few projects proposed different ways to add semantic information to their model. A good example of the different approaches to semantic complexity can be found in [6], who proposed a mixed model using a wide range of different operators, a few of them being related to semantics. Indeed, they used Latent Semantic Analysis (LSA), Latent Dirichlet Allocation (LDA), and a Word2Vec model to extract semantic features.

LSA [9] and LDA [2] are traditional machine learning techniques well-known for their efficiency to extract the topics of a text. Thus, a predictive model of complexity based on either LSA or LDA will detect the main topics of a text and use them to assess complexity. This method is then based on the assumption that some topic are more complex than others. There are, however, two disadvantages with this assumption. First, the model will fit the distribution of topics among the texts from the dataset used to train and test it,

which can be detrimental to its generalizability. Also, we assume here the texts that deal with the same topics are of the same complexity, an hypothesis that may be wrong.

In a different way, words embedding models such as Word2Vec [23] or GloVe [27] learn geometrical encodings of words from their co-occurrence information. Both model achieve to capture the semantics of “analogy”. For instance, computing the difference between vectors of words “king” and “queen”, then adding the vector of “princess” would give a result whose the closest known vector would be the one of the word “prince”. Both models perform well on different kinds of tasks, such as semantic relatedness (to predict the degree of semantic similarity between two words) or concept categorization. These models seem also to provide the best results in capturing semantic complexity [17].

Unfortunately, these techniques project words into an abstract linear space, mathematically meaningful, but hardly understandable. If we can have insights of the vectors’ relationships at the scale of words, as with the example given previously, their interpretations regarding complexity remains limited. At the scale of a text, where we usually compute the average vector of words, we do not know any method to interpret the resulting vector regarding the text complexity.

3. CONCEPTUAL GRAPH PIPELINE

Within the field of semantic analysis, the manipulation of concepts within a text is inherent to its complexity [18]. For instance, when concepts are numerous, abstracts, or not closely related to each other, readers may suffer from accessing their prior knowledge to understand the text [10]. We propose here a concept-based approach of feature engineering to assess semantic complexity.

An ontology (or knowledge graph) is a powerful model to represent concepts and their relationships. With the growth of research in the semantic web, cross-domain description of the world became available, and one of the most known nowadays is DBpedia². DBpedia is a crowd-sourced project to provide an open knowledge graph based on the information available in several Wikimedia³ projects. It provides thus the description in a structured and linked way of various concepts (e.g.: persons, places, organizations, movies, etc.). The English version describes more than 4 million entities so far, and localized versions are provided in 125 languages. Also, everything described in DBpedia is an entity structured through several ontologies (e.g., the DBpedia ontology, schema.org or YAGO). Our work is based on the exploitation of DBpedia and its ontologies to (i) capture conceptual entities from a text, (ii) build a concept graph that includes entities and their higher-order concepts, and (iii) transform the graph into a vector of features. We designed a pipeline to achieve these tasks, whose implementation in Python is open source and publicly available⁴. The pipeline, shown in Figure 1, is built over 5 main components: (i) a text preprocessor, (ii) an entity extractor, (iii) a concept enhancer, (iv) a graph builder and (v) a graph vectorizer.

²<https://wiki.dbpedia.org/>

³<https://www.wikimedia.org/>

⁴<https://github.com/afel-project/pySemanticComplexity>

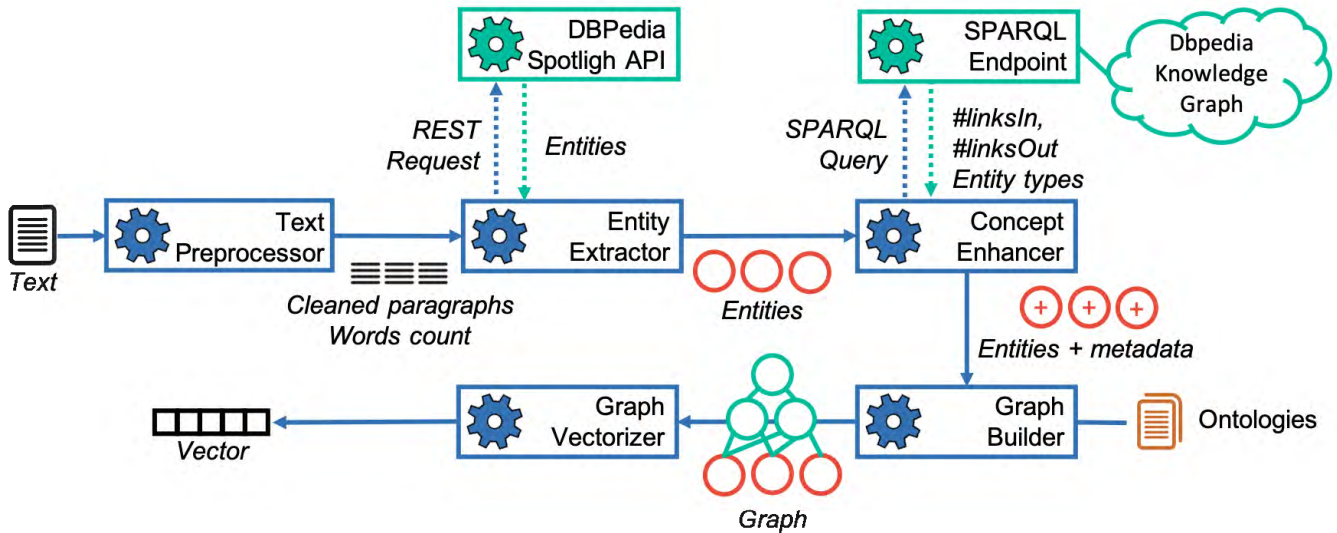


Figure 1: Text to Concept Vector Pipeline.

These software components can be launched separately, either in a stream or batch mode, to support distributing computing. Before we present them in the following subsections, we need to clarify the vocabulary we use to deal with concepts and concept graphs. A concept extracted from a raw text is named **entity**, to follow the vocabulary of DBpedia. An entity is related to some **types**, which are its direct higher-order concepts. Types are classes defined in the ontologies. A **higher-order concept** is a more abstract concept (i.e., a generalization of the concept). Within the ontologies, multiple inheritance relations exist between classes, and allow to structure the concepts they express in term of orders of abstraction. For instance, the “1965 ford mustang” is an entity with a type “collection car”, whose one of the higher-order concept is “car”, generalized itself into “vehicle”, then finally “object”.

3.1 Text preprocessor

Given a text document in a unicode format, this component (i) cleans it, with the deletion of forbidden characters, such as null or backspace, then (ii) splits the text into paragraphs (a prerequisite for the entity extractor that cannot process long text). The text preprocessor also computes the number of words (used for some of the metrics at the end of the pipeline) and the offset for each paragraph (used to locate entities within the whole text).

3.2 Entity Extractor

The entity extractor aims at extracting DBpedia entities from a given list of paragraphs. In order to extract such entities from a text, this component interacts with DBpedia through the REST interface exposed by DBpedia Spotlight⁵, a tool that performs named entity recognition [22]. Given a text, and quality parameters (i.e., prominence, topical pertinence contextual ambiguity and disambiguation confidence), the service returns a list of DBpedia entity URIs with their position in the text and some quality metrics. Thus, from

⁵<https://www.dbpedia-spotlight.org/>

a list of paragraphs, this pipeline stage retrieves a set of DBpedia URI with their positions in the document.

3.3 Concept Enhancer

This stage takes a list of entities and, for each of them, retrieve (i) its related types (second-order concepts), (ii) the number of entities that point to it ($\#linksIn$) and (iii) the number of entities it points to ($\#linksOut$). As we explained before, the entities in DBpedia are represented in a knowledge graph, where entities are linked each other, and described through different ontologies. $\#inLinks$ and $\#outLinks$ are computed based on the relationship between entities in DBpedia.

In order to retrieve all these types for the list of entities, the component interrogates a SPARQL Endpoint. SPARQL (recursive acronym which means SPARQL Protocol and RDF Query Language) is a SQL-like query language to retrieve or manipulate data in the RDF (Resource Description Framework) format, used in DBpedia. The concept enhancer fetches in a first request all the types that are related to each entity of the list. Two others requests are achieved to compute $\#linksIn$ and $\#linksOut$ for each entity. Finally this stage returns the list of entities enhanced with their types and the two basic metrics.

3.4 Graph Builder

The role of the graph builder is twofold: (i) to retrieve the higher-order concepts related to types given along with the entities, and (ii) to build an acyclic graph of all concepts.

The higher order concepts are the super-classes of the types that have been previously retrieved. Indeed, the ontologies used in DBpedia provide a hierarchical structure of classes, where each class may have one or several parent classes. In other words, a concept C may have one or several parents (higher-order concepts), that generalize C and its sibling.

In order to retrieve theses higher-order concepts, the graph builder does not need to interact with any DBpedia end-

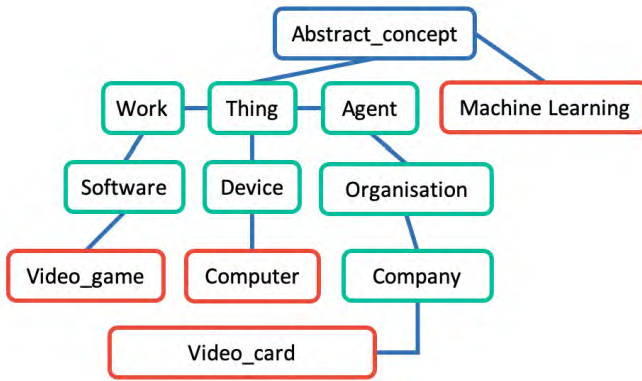


Figure 2: Example of a Concept Graph.

point. This component loads directly the structure of ontologies, which are RDF files that can be locally processed. In our pipeline, we exploit the three most used ontologies in DBpedia: the DBpedia ontology, schema.org and YAGO. For each given type that belongs to one of the three ontologies, the graph builder recursively retrieves its parent concepts. As a result, the component constructs a set of pairs $\langle \text{child concept}, \text{parent concept} \rangle$ and, with the given list of entities, builds a concept graph of the text.

This graph is an acyclic graph composed of two kinds of vertices: entities (that appear in the text) and higher-order concepts (classes from the ontologies). An entity vertex has the following attributes: (i) the DBpedia URI, (ii) the list of positions in the text (word index) (iii) the $\#linksIn$ value and (iv) the $\#linksOut$ value.

While an entity may appear several times in the text, it is represented as a single vertex in the graph. However, its different positions are recorded in its attributes. The other vertices (higher-order concepts) contain only their URI. In order to ensure the graph is connected (a mathematical property required to compute some of the features explained in the following section), an abstract highest-order vertex is inserted and linked to any vertex that does not have any higher-order concept.

We illustrate in Figure 2, an example of a graph computed from the text “This computer has a powerful graphic card that is suitable for video games and machine learning.”. Entity nodes are colored in red, while higher-order concepts are in green (the abstract highest-order concept is in blue). In this example, the Spotlight API extracted 4 entities from the DBpedia ontology: “video game”, “computer”, “video card” and “machine learning”. The green nodes on the graph are then classes of this ontology that generalize the concepts they are connected to. Thus, a video game is a software, which is a work, whose generalized concept is a thing.

3.5 Graph Vectorizer

Giving the graph, and the number of words computed in the first stage of the pipeline, this last component applies several calculus in parallel to produce 17 metrics. The values are embedded in an vector, which is the final output of the pipeline. This vector can be used afterwards as an input for a predictive model on conceptual complexity.

4. CONCEPT COMPLEXITY METRICS

In this section, we explain the metrics that highlight specific information about the conceptual complexity, based on the structure of the generated graph. The 17 metrics computed by the pipeline are the following: (i) $\#concepts$, (ii) $\#distinctConcepts$, (iii) $conceptsByWords$, (iv) $distinctConceptsByWords$, (v) \muTypes , (vi) \sigmaTypes , (vii) $\mu linksIn$, (viii) $\sigma linksIn$, (ix) $\mu linksOut$, (x) $\sigma linksOut$, (xi) $\#nodes$, (xii) $Radius$, (xiii) $Diameter$, (xiv) $Density$, (xv) $Assortativity$, (xvi) $\mu TextDensity$ and (xvii) $\sigma TextDensity$. We details them in the following subsections.

4.1 Basic Concept Metrics

The first four metrics are measures based on the entities extracted from the text. $\#concepts$ is the total count of concepts the entity extractors retrieved, while $\#distinctConcepts$ is the number of the different concepts extracted. These metrics are based on the assumption that the more concepts a text deals with, the more complex it is.

Since these features might be correlated to the size of the text (which, as explained later, was not the case in our experiments, but still might be), we also propose **concepts-ByWords** and **distinctConceptsByWords**, the ratio between $\#concepts$ and $\#distinctConcepts$ respectively, and the number of words.

4.2 Concept Connection Metrics

The six following indicators relate the direct properties of the concepts that appear in the text. \muTypes and \sigmaTypes are respectively the mean and standard deviation of the number of types per entity. As explained before, each concept extracted from the text is linked to its types (i.e. its direct higher-order concepts). We suppose here that the more an entity has types, the more concepts of higher-order are required to explain it, and thus the more complex this entity is. We use the two basic statistic descriptors to capture that notion at the scale of a document.

For each entity in our graph e , we also have $\#linksIn$, the number of links that go from entities of the global DBpedia knowledge graph to e , and $\#linksOut$, the number of links that go from e to others entities of that same graph. We compute then two statistic descriptors for each indicator at the document level. $\mu linkIn$, $\sigma linkIn$ are respectively the mean and standard deviation of the number of entities in DBpedia that point to the entities of the document, while $\mu linkOut$ and $\sigma linkOut$ are about the entities in DBpedia that are pointed by the entities of the document. We suppose here that the more relations an entity has with others, the more popular it is, and the less complex it might be.

4.3 Concept Abstraction Metrics

The next five metrics take into account the whole concept graph: the entities, their types and the higher-order concepts retrieved recursively from the ontologies.

$\#nodes$ is the total number of nodes in the graph. The higher it is, the more concepts have been used or the more specific they are (i.e., the more higher-order concepts there are to specify them).

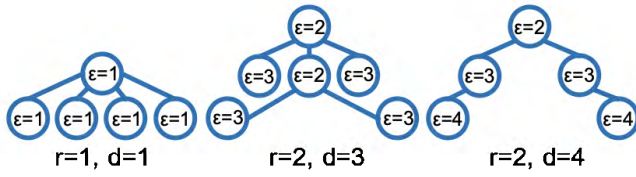


Figure 3: Examples of radius and diameters.

For the next metrics a formalism is required. Let $G = (V, E)$ the definition of a graph G with V the set of vertices (the concepts) and E the set of edges. Let $\forall(u, v) \in V^2$, $d(u, v)$ the distance between the nodes u and v : the number of edges in the shortest path. Since our graph is connected (thanks to the abstract higher-order concept node) and undirected, we have the following properties:

$$\forall(u, v) \in V^2, d(u, v) \geq 1, d(u, v) = d(v, u) \quad (1)$$

Interpreted in our context, the distance between 2 concepts shows how close they are from each other. Since concepts are linked by their higher-order concepts, the distance is the length of the path from one concept c_1 to another concept c_2 , going through their closest common higher-order concept. Thus, the distance measures how much we have to generalize to find the common inherent concept to relate c_1 and c_2 .

ϵ is the eccentricity of a vertex v . It is defined as the greatest distance between v and any other vertex of the graph.

$$\forall v \in V, \epsilon(v) = \max_{u \in V} d(v, u) \quad (2)$$

In our context, the eccentricity for a concept c_1 gives a measure of how far this concept is from the others.

4.3.1 Radius and Diameter

Based on this eccentricity, the **radius** r_G is the minimum eccentricity of any vertex in the graph.

$$r_G = \min_{v \in V} \epsilon(v) = \min_{v \in V} [\max_{u \in V} d(v, u)] \quad (3)$$

At the opposite, the **diameter** of a graph d_G is the maximum eccentricity of any vertex in the graph.

$$d_G = \max_{v \in V} \epsilon(v) = \max_{v \in V} [\max_{u \in V} d(v, u)] \quad (4)$$

On the one hand, the diameter highlights the “spreadness” of concepts: the more unrelated and specific concepts we have, the higher the diameter will be. On the other hand, the radius points to the “compactness” of concepts out: the more the concepts are closely related to each other, the lower the radius will be. Note that the radius is not strictly the opposite of the diameter; both metrics measure information that are not theoretically linearly dependent. To have an insight of their meaning, Figure 3 shows different simple graph structures with their radius and diameter.

4.3.2 Density

The **density** of a graph stresses how much nodes are connected to each other. A graph is dense if the number of edges is close to the maximal possible number of edges. The opposite is a sparse graph, that has few edges. The density is computed by the following equation:

$$0 \leq \mathcal{D}_G = \frac{2|E|}{|V|(|V| - 1)} \leq 1 \quad (5)$$

Here, a concept graph with a high density implies that concepts and higher-order concepts are closely related to each other. The text may deal with many different concepts, but many of them share parents concepts and so on. Density may then be a factor of discrimination regarding two texts that have similar numbers of concepts, but with one using concepts from a unique domain while the other one deals with varied concepts.

4.3.3 Assortativity

Following Newman [26], the **assortativity** measures in a graph the similarity of connections with respect to the vertex degree. The degree is the number of connections (the number of edges) a vertex has to other vertices. Mathematically, the assortativity is the Pearson correlation coefficient of degree between all pairs of vertices. It is computed by the following equation:

$$\mathcal{A}_G = \frac{2 \sum_i j_i k_i - \frac{1}{2} |E|^{-1} (\sum_i j_i + k_i)^2}{\sum_i (j_i^2 + k_i^2) - \frac{1}{2} |E|^{-1} (\sum_i j_i + k_i)^2} \quad (6)$$

where j_i, k_i are the degrees of the vertices at the ends of the i^{th} edge, in a graph with $|E|$ edges.

Because \mathcal{A}_G is a correlation coefficient, it lies between -1 and 1 . When it is close to 1 , the graph shows a perfect assortative mixing: the vertices in the network that have many connections tend to be connected to the other vertices with many connections. When \mathcal{A}_G is close to -1 , the graph is disassortative: the nodes that have many connections tend to be linked to the nodes with few connections. When \mathcal{A}_G is close to 0 , the graph is non assortative: there is no particular correlation between node connections and their degree.

Because in our concept graphs, concepts (vertices) are connected only by their relationship of abstraction (a parent concept being a more general concepts), a positive assortativity would signify that the more parents a concept requires to be defined, the more grand-parents these parents have. Under the hypothesis that the more higher-order concepts we need to define a concept c , the more complex c is, the assortativity may be a candidate metric to evaluate complexity: a graph with a high assortativity may highlight a text that deals with very complex and precise concepts.

Compared to the μ Types indicator, which is only computed on the basis of entity types, this metric takes into account the whole graph. For instance, if the text uses very narrowed concepts, that have only few types, which are however defined by many higher-order concepts, μ Types would be low, but assortativity may be close to 1 .

4.4 Concept Organization Metrics

While the previous metrics are about the graph only, they do not take into account the positions of the different entities

within the text. Although they provide conceptual information about the overall document, they lack giving insight about the evolution of complexity at a local level. For instance, a paragraph that deals with closely related concepts may be simpler than a paragraph that manipulates different unrelated concepts. Thus two texts that handle similar concepts, but with a different structure, may have a different complexity. In order to capture such information, we have built a metric based both on the concept graph and on the positions of the entities within the text.

Let $N \subset V$ the subset of vertices that contains the entities. As we explained in the previous section, the entities include the list of the positions of their occurrences. Let $occ(e)_i \forall e \in N$ the position of the i^{th} occurrence of the entity e in the text. We define the function sp of two entities that returns the shortest path in the text between the two concepts.

$$\forall(n, m) \in N^2, sp(n, m) = \min_{i,j} |occ(n)_i - occ(m)_j| \quad (7)$$

On the basis of sp , we define the textual density td between two entities as the product of their graph distance d and the opposite of their textual distance td , normalized by the product of the graph diameter D_G and the text length L_T .

$$\forall(n, m) \in N^2, td(n, m) = \frac{d(n, m)}{D_G} \cdot \frac{1 - sp(n, m)}{L_T - 1} \quad (8)$$

Since $0 \leq d(n, m) \leq D_G$ and $0 \leq sp(n, m) \leq L_T - 1$, we have $0 \leq td(n, m) \leq 1$. When td is near 0, the concepts are either semantically very close, or unrelated and far to each other in the text. When td is close to 1, the concepts are semantically far to each other but appear closely in the text. At the document scale, we compute then the two last metrics $\mu\text{TextDensity}$ and $\sigma\text{TextDensity}$ that are respectively the mean and standard deviation of the textual density of all pairs of entities.

5. CONCEPTUAL COMPLEXITY ASSESSMENT EVALUATION

In this section, we seek to evaluate the performance of a classifier that predicts complexity, based on the concept metrics defined above. In order to analyze the impact of our features on such predictive models, we carried out several rounds of evaluation. We started building a binary classifier based on the state-of-the-art features to obtain a model of reference. To answer our first research question, “Could a model based on our sole metrics outperform a state-of-the-art predictive model of semantic complexity?”, we trained another classifier that uses our conceptual complexity features only and compared it with the first one.

Afterwards, we considered the second question: “Does the extension of a state-of-the-art predictive model with our engineered features improves its performance?”. To compare two models, one being an extension of the other, we trained our two models on the same splits of data (within a cross-validation procedure) and compared the two lists of mea-

sures of performance with a statistical test to assess whether the performances of the extended model was significantly higher than the performances of the first one.

Finally, since results were positive with respect to question 2, we trained a last classifier based on syntactic, lexical and semantics features, to evaluate how well it could perform for the specific task of predicting the overall complexity of learner’s productions.

5.1 Dataset

All models here were trained on an extract of the EF-Cambridge Open Language Database [16, 13]. This database is a text corpus of documents written by adult learners of English as a foreign language. In this study, we use a subset of this database, that includes 41 626 essays.

Human examiners evaluated these essays in order to assess the learners’ level of knowledge defined in the global scale provided in the Common European Framework of Reference for Languages⁶. This scale define 6 levels of knowledge (i.e.: A1, A2, B1, B2, C1 and C2), that describe skills in reading, listening, speaking and writing. Regarding this last domain of competency, A1 and A2 levels target beginners: they can interact in a simple way, using familiar everyday expressions and very basic phrases. Independent users belong to the B level group. Compared to level groups A and C, this group presents a clear difference between its two levels. While B1 users can produce simple connected texts on topics that are familiar, B2 users are able to write clear and detailed text on a wide range of topics and give their point of view on a topical issue. Finally, C1 and C2 levels target proficient users, that can handle complex subjects and produce clear, well-structured and detailed texts.

Thus, in this dataset, examiners labeled learners’ essays with one of these levels. Although there is no formal description about the evaluation process, the definitions provided in the global scale of the framework relate to the different kind of complexity we exposed earlier (i.e.: syntactic, lexical and semantic). For this study, we define two categories of writings based on these labels. The documents evaluated with a level between A1 to B1 are considered in the G1 group, while the documents evaluated with a level between B2 to C2 belong to the G2 group.

Outliers were removed from the dataset. They were defined here as the documents where the number of words were below the first centile or above the last one, or where the number of concepts were again, below the first or above the last centile. Since the distribution of documents between G1 and G2 was skewed, we reduce the dataset to obtain an equal proportion of samples in each group. The baseline score of a binary classifier is then 0.5 for accuracy, precision or recall.

Also, we detected a potential bias in the dataset, as the samples in G2 tend to be longer (in numbers of words) than the ones in G1. To prevent our models from reproducing that bias, we filtered all conceptual features we explained previously that would present a significant correla-

⁶<https://www.coe.int/en/web/common-european-framework-reference-languages/level-descriptions>

tion ($r \geq 0.75$, $p\text{-value} < 0.003$) with the length of the text (p-value was computed using the Bonferroni correction since we computed different statistical tests, which raise then the risk to produce a statistical type I error). It appears that none of the 17 features were significantly correlated to the length of the text as much as this threshold. Finally, we examine the potential relationship between pairs of features, to assess their independence and to avoid training our models with correlated features. We looked for strong significant correlations: $r \geq 0.9$ with $p\text{-value} \leq 0.0004$ (again, p-value was computed using the Bonferroni correction). It appears that only #concepts and #distinctConcepts are strongly correlated ($r = 0.936$). We then decided to remove the latest feature from our models for this dataset.

5.2 Baseline Model Evaluation

In order to produce a baseline binary classifier of semantic complexity, we applied the following methodology: (1) compute a 300 dimensional average vector for each sample based on pre-trained Glove vectors ; (2) train different types of classifiers using 5-folds cross validation, with default hyperparameters for each of them; (3) select the model that provides the best f1 score; (4) split the whole dataset into one subset for hyper parameters tuning (90%), and one subset for final testing (10%); (5) tune the hyper parameters of this model using a grid search 10-folds cross validation and (7) test the final model on the last subset.

Glove is an unsupervised machine learning algorithm used to compute vector representation for words, proposed by Stanford University [27]. Based on the word-to-word co-occurrence statistics obtained from a corpus, GloVe computes a representation of words into a vector space. In our case, we did not train such a model from scratch on our dataset, but used pre-trained words vectors available online⁷. These vectors were computed from the 2014 corpus of Wikipedia and the 5th edition of English Gigaword⁸, in a 300 dimensions space. For each sample from our dataset, we computed the mean of the tokenized words vectors, using 0 padded vectors for unknown words. The tokenization process was achieved using the NIST Tokenizer⁹.

The main scoring metric we used to evaluate our models is the f1-score. This metric takes in account both precision and recall through the following formula: $2 \cdot P \cdot R / (P + R)$, given the precision P (the ratio between the number of true positives and the sum of the number of true positives and false positives) and the recall R (the ratio between the number of true positives and the the sum of the number of true positives and false negatives). This metric is a value between 0 (the worst) and 1 (the best).

In step 2, we tested 10 different models. Each model was a pipeline made of two stages: a pre-processing stage to standardize features by removing the mean and scaling to unit variance, and the classifier stage. The results for the classifiers selection are shown in Table 1. The best classifier was the multilayer perceptron (MLP); it achieved the best f1 score, but also the best average accuracy. MLP is a type of

Table 1: Glove-based Models Performances.

Classifier	μ F1	μ Accuracy
MLP (1 hidden layer of size 100)	0.936	0.935
SVC with rbf kernel	0.936	0.934
K-nearest neighbors	0.917	0.910
SVC with polynomial kernel (d=2)	0.916	0.911
Quadratic discriminant	0.907	0.905
SVC with linear kernel	0.899	0.897
Random Forest (100 estimators)	0.896	0.892
AdaBoost	0.865	0.862
Gaussian Naive Bayes	0.826	0.811
Decision Tree	0.811	0.810

feedforward neural networks composed of at least one hidden layer of nodes, where each node in the hidden and output layers uses a nonlinear activation function [14].

In the last step, we tuned the following hyperparameters of the MLP model, given with their set of evaluated values: **(i)** the activation function (logistic, hyperbolic tangent or Rectified Linear Unit), **(ii)** the regularization term α (0.0001, 0.001, 0.01, 0.1 or 1.0), **(iii)** the size of each hidden layer (100 or 200), **(iv)** the number of hidden layers (1 to 5), **(v)** the learning rate (constant or adaptive) and **(vi)** the solver used for the weight optimization (adam or lbfgs).

We tested the different combinations of hyperparameters with a grid search 10-folds cross validation on a stratified subset of 90% of our dataset, using the mean test f1 value as the scoring method. The best configuration appears to be a MLP with one hidden layer of 200 nodes using a ReLU (Rectified Linear Unit) activation function, an α value of 0.01 with an adaptive learning rate and an adam solver.

Table 2: Test Scores of the Baseline and Concept Based Models.

Score	Baseline M. Values	Concept M. Values
F1	0.937	0.888
Accuracy	0.939	0.889
Precision	0.982	0.894
Recall	0.896	0.882

Eventually, we tested that classifier on the remaining 10% of our dataset. The different metrics are exposed in the column “Baseline M. Values” of Table 2. The Precision-Recall curve is illustrated in Figure 4a. Overall, this baseline classifier shows better precision than recall. As we can see on Figure 4a, recall tends to drop quickly as precision goes over 0.9. Recall is defined as the ability for the classifier to avoid false negatives. In our context, recall is thus the ability for the model to avoid predicting a text as being simple while it is in reality complex. In other terms, this models is better at predicting a text as being complex, than predicting a text as being simple.

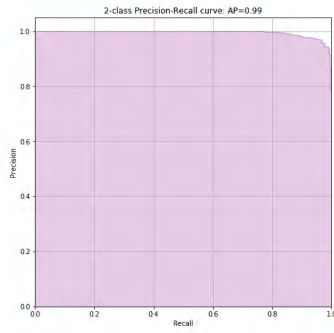
5.3 Concept Based Model Evaluation

On the basis of the pipeline architecture elicited previously, we trained and tested an equivalent model that works with our 16 elicited engineered features only. The model is then composed of a standardization stage and an MLP classifier

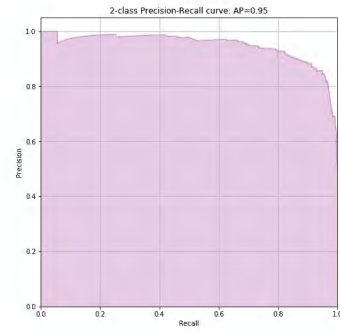
⁷<http://nlp.stanford.edu/data/glove.6B.zip>

⁸<https://catalog.ldc.upenn.edu/LDC2011T07>

⁹http://www.nltk.org/_modules/nltk/tokenize/nist.html



(a) Baseline Model.



(b) Concept Features-Only Model.

Figure 4: Precision-Recall Curves of Baseline and Concept Features-Only Models.

with the previous selected hyperparameters. This model was trained in a stratified 10-folds cross validation on 90% of the dataset and tested with the remaining 10%. The results are given in the third column of Table 2. The Precision-Recall curve is illustrated in Figure 4b.

Using only the conceptual graph-based features, we can see that this model is worst than the baseline one to predict complexity. The drop of score value is around 5% for F1 and accuracy, and 8% for the precision. In conclusion, to answer to our first research question, using the features we proposed only, we cannot outperform a model based on the state-of-the-art semantic features to predict complexity.

5.4 Baseline and Mixed Model Comparison

Whereas the model based on conceptual features failed to outperform the baseline classifier, these metrics might still add information that are not present from the state-of-the-art semantic features. Thus, we need to study whether a classifier would do better with both sets of features.

In order to assess such a result, and because the two classifiers will not be independent, we need to evaluate whether the performance scores of these classifiers have a significant difference of mean. We trained two classifiers, CL_{base} and CL_{mixed} with the same architecture than the one selected previously: a standardization pre-processing stage chained to a MLP classifier with the former chosen hyperparameters. CL_{base} is the baseline classifier that uses the Glove features only, while CL_{mixed} uses both Glove and our conceptual graph-based features. In order to compare the difference of performances, we evaluated our classifiers with the same splits of a stratified 10-fold cross validation scheme.

The results for the 4 metrics (F1, accuracy, precision and recall) for CL_{base} and CL_{mixed} on each fold are given in Table 3. The metrics observed for CL_{mixed} seem to be better than CL_{base} . However, to assess whether this difference is significant, we proceed a paired sample Student's T-test, since the measurements of each classifier were applied on the same splits of data.

The Table 4 presents the statistical results of the paired T-test. The null hypothesis is that the pairwise difference between the two tests for each metrics is equal. With the degrees of freedom $df = 9$ (as we achieved 10 measures),

and a p-value of 0.05, the t-table value is 1.812. For each metrics, we can see on Table 4 than the computed t-value has a absolute value above 1.812. We can then rejected the null hypothesis that there is no difference between means for each metrics. In conclusion, adding conceptual graph-based features to Glove features improve significantly the performance scores of our complexity classifier. We therefore answer positively to our second research question.

5.5 Extended Complexity Classifier

Thus, conceptual graph-based complexity features can increase the performances of complexity classifiers, while improving the explainability of the model. In our context of evaluating learners' production in English to assess their complexity, we eventually would like to test how a complete classifier, using not only semantic features but also syntactic and lexical ones, would perform.

In that last part, we trained and tested a supervised classifier that predicts complexity based on the previous semantic features, and syntactic and lexical features proposed in [20], that appears to be the state-of-the-art metrics so far. We removed the ones strongly correlated with the length of the text, as our dataset presents a potential bias of categorical distribution over the text length. We ended up with 30 syntactic and lexical features, 300 features from Glove and 17 conceptual features. The classifier has the same architecture as those trained before.

The results for the test are presented in table 5, while the Precision-Recall curve is illustrated in Figure 5. With high scores in both precision and recall, this model seems to be suitable to assist teachers in their assessment of complexity.

6. CONCLUSION AND PERSPECTIVES

At the core of several learning related activities, the assessment of complexity is a task of importance. This topic has been broadly studied through the structural analysis of texts, with the design and evaluation of lexical, syntactic or morphological measures. The consideration of semantic metrics, however, is still scarce. Recently, word embedding techniques have demonstrated their promising potential to design powerful predictive models of semantic complexity, but lack explainability. In order to overcome this obstacle, we proposed an approach based on the exploitation of existing knowledge graphs to generate a graph representa-

Table 3: Comparative Scores of Baseline and Mixed Classifiers.

CL_{base} scores				CL_{mixed} (Mixed) scores			
F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.
0.943	0.943	0.940	0.947	0.954	0.954	0.958	0.949
0.950	0.950	0.946	0.955	0.962	0.962	0.966	0.957
0.938	0.937	0.926	0.951	0.947	0.946	0.934	0.960
0.947	0.947	0.937	0.957	0.952	0.951	0.945	0.959
0.947	0.947	0.933	0.963	0.953	0.953	0.952	0.953
0.941	0.940	0.930	0.952	0.946	0.946	0.937	0.956
0.944	0.945	0.945	0.944	0.955	0.955	0.951	0.959
0.947	0.947	0.939	0.955	0.958	0.958	0.956	0.960
0.954	0.954	0.952	0.956	0.965	0.965	0.965	0.964
0.941	0.940	0.920	0.964	0.956	0.955	0.947	0.964

Table 4: Paired T-test Results between Scores of Baseline and Mixed Classifiers.

Scores	Computed t-value
F1	-9.230
Accuracy	-9.271
Precision	-6.303
Recall	-1.922

Table 5: Test Scores of the Extended Complexity Classifier.

Score	Value
F1	0.970
Accuracy	0.969
Precision	0.956
Recall	0.984

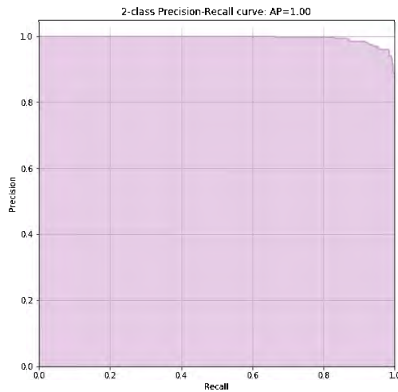


Figure 5: Precision-Recall Curve of the Extended Complexity Classifier.

tion of a text, whose concepts are exposed and related each other through their abstractions. We suggested 17 metrics computed from this concept graph to highlight intelligible information about the semantic complexity of the text.

We evaluated our proposition within the context of learners' productions for a European certification of English as a foreign language. We observed that a word embedding based classifier still tends to surpass a model relying solely on our features. Nevertheless, a classifier that uses these two sets of features together outperforms significantly the previous ones. At last, we proposed a classifier using these semantic metrics but also syntactic and lexical features. Tested in our context of learners' evaluation, its performances seem to make it suitable to assist human examiners in their tasks.

Eventually, we found, at the time of writing, a similar approach to ours, based on the DBpedia knowledge graph to measure conceptual complexity [30]. Authors used concepts extraction in the context of text simplification. Although not based on a concept graph as we did, the metrics they proposed may be also relevant for our domains of application. We will then integrate them to our model shortly. While first results obtained are promising, we have to dig into the analysis of each metric we suggested, in order to evaluate their power of discrimination regarding the conceptual complexity of the features we proposed. At a longer term, it will offer opportunities to consider a proactive usage on users to assist them in their learning activities. Assessing complexity is also of importance for learning resources indexing, to provide useful recommendations. Since this domain of application is different from the present context of study, we will consolidate a dataset of learning objects and then reproduce the experimentation to evaluate how our approach generalizes to other tasks.

7. REFERENCES

- [1] O. Biran and K. McKeown. Human-Centric Justification of Machine Learning Predictions. In *Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 1461–1467, California, 2017. International Joint Conferences on Artificial Intelligence Organization.
- [2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

- [3] V. Butoiianu, P. Vidal, E. Duval, J. Broisin, and K. Verbert. User Context and Personalized Learning: a Federation of Contextualized Attention Metadata. *Journal of Universal Computer Science*, 16(16):2252–2271, 2010.
- [4] C. Cechinel, S. S. Alonso, M.-Á. Sicilia, and M. C. de Mattos. Descriptive Analysis of Learning Object Material Types in MERLOT. In *Research Conference on Metadata and Semantic Research*, pages 331–341. Springer, 2010.
- [5] E. Dale, J. C. E. English, and 1949. The concept of readability. *Elementary English*, 26(1):19–26, 1949.
- [6] M. Dascalu, P. Dessus, S. Trausan-Matu, M. Bianco, and A. Nardy. ReaderBench, an Environment for Analyzing Text Complexity and Reading Strategies. In *International Conference on Artificial Intelligence in Education*, pages 379–388. Springer, 2013.
- [7] M. Dascalu, G.-M. Gutu, S. Ruseti, I. C. Paraschiv, P. Dessus, D. S. McNamara, S. A. Crossley, and S. Trausan-Matu. ReaderBench - A Multi-lingual Framework for Analyzing Text Complexity. In *European Conference on Technology Enhanced Learning*, pages 495–499. Springer, 2017.
- [8] E. Davoodi and L. Kosseim. On the Contribution of Discourse Structure on Text Complexity Assessment. *arXiv.org*, Aug. 2017.
- [9] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.
- [10] C. A. Denton, M. Enos, M. J. York, D. J. Francis, M. A. Barnes, P. A. Kulesz, J. M. Fletcher, and S. Carter. Text-Processing Differences in Adolescent Adequate and Poor Comprehenders Reading Accessible and Challenging Narrative and Informational Text. *Reading Research Quarterly*, 50(4):393–416, Oct. 2015.
- [11] J. Falkenjack and A. Jönsson. Classifying easy-to-read texts without parsing. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*, pages 114–122, 2014.
- [12] J. Falkenjack, K. H. Mühlenbock, A. J. P. o. t. 19th, and 2013. Features indicating readability in Swedish text. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013)*, pages 27–40, 2013.
- [13] J. Geertzen, T. Alexopoulou, and A. Korhonen. Automatic linguistic annotation of large scale l2 databases: The ef-cambridge open language database (efcamdat). In *Proceedings of the 31st Second Language Research Forum. Somerville, MA: Cascadilla Proceedings Project*, 2013.
- [14] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.
- [15] Q. Han and F. Gao. Towards semantic learning object metadata: mapping standard metadata specifications to ontologies. In *Proceedings of IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE) 2012*, pages H1C–12. IEEE, 2012.
- [16] Y. Huang, A. Murakami, T. Alexopoulou, and A. Korhonen. Dependency parsing of learner English. *International Journal of Corpus Linguistics*, 23(1):28–54, May 2018.
- [17] Z. H. Kilimci and S. Akyokus. Deep Learning- and Word Embedding-Based Heterogeneous Classifier Ensembles for Text Classification. *Complexity*, 2018(7):1–10, 2018.
- [18] W. Kintsch, T. A. Van Dijk Psychological review, and 1978. Toward a model of text comprehension and production. *Psychological review*, 85(5):363, 1978.
- [19] B. Kopainsky, P. P. Dummer, and S. M. Alessi. Automated assessment of learners’ understanding in complex dynamic systems. *System Dynamics Review*, 28(2):131–156, Apr. 2012.
- [20] X. Lu. Automatic analysis of syntactic complexity in second language writing. *International journal of corpus linguistics*, 15(4):474–496, 2010.
- [21] X. Lu. Automated measurement of syntactic complexity in corpus-based l2 writing research and implications for writing assessment. *Language Testing*, 34(4):493–511, 2017.
- [22] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems*, pages 1–8. ACM, 2011.
- [23] T. Mikolov, K. C. 0010, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*, 2013.
- [24] D. Napolitano, K. Sheehan, and R. Mundkowsky. Online Readability and Text Complexity Analysis with TextEvaluator. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 96–100, 2015.
- [25] S. Narayan and C. Gardent. Hybrid simplification using deep semantics and machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 435–445, 2014.
- [26] M. E. Newman. Assortative mixing in networks. *Physical review letters*, 89(20):208701, 2002.
- [27] J. Pennington, R. Socher, and C. D. Manning. Glove - Global Vectors for Word Representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [28] M. T. Ribeiro, S. Singh, and C. Guestrin. ”Why Should I Trust You?”. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM, 2016.
- [29] A. Siddharthan. A survey of research on text simplification. *ITL-International Journal of Applied Linguistics*, 165(2):259–298, 2014.
- [30] S. Stajner and I. Hulpus. Automatic assessment of conceptual text complexity using knowledge graphs. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 318–330, 2018.

Affective State Prediction in a Mobile Setting using Wearable Biometric Sensors and Stylus

Rafael Wampfler
Dept. of Computer Science
ETH Zurich, Switzerland
wrafael@inf.ethz.ch

Severin Klingler
Dept. of Computer Science
ETH Zurich, Switzerland
kseverin@inf.ethz.ch

Barbara Solenthaler
Dept. of Computer Science
ETH Zurich, Switzerland
solenthaler@inf.ethz.ch

Victor R. Schinazi
Dept. of Humanities, Social
and Political Sciences
ETH Zurich, Switzerland
scvictor@ethz.ch

Markus Gross
Dept. of Computer Science
ETH Zurich, Switzerland
grossm@inf.ethz.ch

ABSTRACT

The role of affective states in learning has recently attracted considerable attention in education research. The accurate prediction of affective states can help increase the learning gain by incorporating targeted interventions that are capable of adjusting to changes in the individual affective states of students. Until recently, most work on the prediction of affective states has relied on expensive and stationary lab devices that are not well suited for classrooms and everyday use. Here, we present an automated pipeline capable of accurately predicting (AUC up to 0.86) the affective states of participants solving tablet-based math tasks using signals from low-cost mobile bio-sensors. In addition, we show that we can achieve a similar classification performance (AUC up to 0.84) by only using handwriting data recorded from a stylus while students solved the math tasks. Given the emerging digitization of classrooms and increased reliance on tablets as teaching tools, stylus data may be a viable alternative to bio-sensors for the prediction of affective states.

Keywords

Classification, Affective Computing, Stylus, Biometric Sensors

1. INTRODUCTION

Affective states are psycho-physiological constructs used to characterize the emotions (short-lived) and moods (long-lived) that arise and are experienced while individuals are engaged with a stimulus. Affective states play an important role in the educational context and can directly influence a student's learning gain [9, 26, 10]. For example, learning outcomes have been found to decrease if frustration is persistent during problem solving, whereas overcoming a state of frustration can have a positive effect on learning [10].

Previous research has investigated the relationship between affective states and learning performance by attempting to detect the diverse emotions that occur during learning. The logic behind this approach is that, depending on the emotion of a student, appropriate actions can be taken in order to assist students during learning (e.g., adapting task elements in the case of intelligent tutoring systems (ITS) and self-regulation by providing affective feedback).

Previously, a wide range of data sources have been used to measure and predict affective states in the learning context including audio and video [32], interaction data [21, 14] and bio-sensors [8, 4]. Systems that rely on the analysis of audio (e.g., speech) and video (e.g., facial expression) data [32] cannot guarantee full anonymity and are subject to privacy issues. Given these limitations, researchers have attempted to derive the affective state of individuals based on interaction data which contain log data of the user's interaction with the learning system, such as input and error behavior, timing and help calls [21, 14]. Although large and powerful interaction data sets can be easily collected especially in online environments, the features are typically dependent on the learning domain and on the specific learning system. Attempts towards a cross-domain or cross-system engagement model have been presented (e.g., for learning spelling and math [18]), but these generalized methods typically have a lower accuracy as domain-specific features. Data from bio-sensors (e.g., measuring muscle activity [8] and heart rate [4]) have also been used to predict emotions. However, most of these devices are typically restricted to lab settings, expensive and difficult to operate, and somewhat intrusive. Recently, a variety of portable and low-cost bio-sensor devices have become available (e.g., Shimmer GSR+ and Polar H10). These devices have the potential to transform education research because they can be used to monitor a learner's physiological state at home or in a classroom.

In this paper, we explore a low-cost mobile setup to detect the affective state of students. Our goal is a system to detect affective states that is cheap and easy to operate, can be used outside a lab setting, is non-intrusive, and minimizes potential issues related to privacy. We consider bio-sensor data from skin conductance, heart measures, and skin temperature. In addition, and in contrast to previous work in

Rafael Wampfler, Severin Klingler, Barbara Solenthaler, Victor Schinazi and Markus Gross "Affective State Prediction in a Mobile Setting using Wearable Biometric Sensors and Stylus" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 198 - 207

the field of learning systems, we also evaluate handwriting data recorded by a stylus to predict the affective state of students. Here, we use the fact that tablets bundled with a stylus are becoming increasingly available in households and classrooms and are inherently non-intrusive and mobile.

We propose a generic pipeline in which we process the data from the bio-sensors and stylus in order to extract a set of features for each of the sensors. We then use a classification model to predict the current affective region in the valence-arousal space of emotions [29]. Valence describes how much an emotion is perceived as positive or negative and arousal represents the intensity of the emotion. Our method allows researchers to define arbitrary areas of interest in the valence-arousal space, and can be applied to a wide range of applications and questions of interest. We evaluated our method by applying it to a math problem solving scenario in which participants provided answers in unstructured handwriting on a tablet device. Best performance was reached when data from all sensors was used for prediction (0.88 AUC). Interestingly, we reached a comparable performance using only the data acquired by the stylus (0.84 AUC). These results suggest that a simple tablet with a stylus can be sufficient to reliably predict a student's emotional state. Finally, we also explored whether the affective state model could be generalized over domains. For this purpose, we applied the trained model to a passive setting with picture stimuli leading to a performance of 0.68 AUC.

2. RELATED WORK

Affective States and Interaction Data. Due to their influence on learning gain, affective states play an important role in education in general and in particular during math learning [27, 21]. Boredom was shown to negatively influence the learning gain [9, 26], while engaged concentration can improve the learning outcome [9]. Interestingly, frustration and confusion can positively affect learning in case the student is able to resolve these states [10]. One line of research tries to predict these affective states based on logged user interactions only. Frustration, boredom, engaged concentration and confusion have been successfully predicted using interaction data for math tutoring systems [21, 14]. On the other hand, valence and arousal have been predicted using mouse and keyboard interaction data from writing compositions in free text [31]. Moreover, generalized models have been proposed, such as an engagement model for two different learning domains and tutors (spelling and math) [18]. Based on such automatically predicted affective states, different intervention strategies have been explored. An automatic student-centered affect-aware feedback loop was shown to increase the learning gain [14] while other work explored how teachers can provide better interventions based on real-time information about the evolution of student's affective states [11].

Biometric Sensors. Biometric sensors provide an objective measure of the physiological reactivity of users engaging with a learning environment while minimizing interference with the actual task [19, 4, 17, 30]. Indeed, educational research has investigated the effectiveness of a variety of physiological signals used to infer affective states. Electrodermal activity, skin temperature, and heart rate were generally found to be good predictors of emotions [19, 17,

30] and mind wandering [4] across different tasks including math learning [17, 30], scientific text reading [4] and audio, visual and cognitive stimuli in general [19]. However, these previous works mainly focused on expensive, high quality sensors to provide medical grade accuracy for the measurement of physiological signals. In contrast, we focus on an affective tutor that can be used in learning systems, hence we gather such data in a non-intrusive and easy to use way.

Stylus. Predicting affective states based on stylus data is still a relatively new research topic. Likforman-Sulem et al. [24] predicted anxiety, depression and stress based on figure drawings and writing given words. Fairhurst et al. [12] conducted an experiment for predicting stress and happiness by letting participants writing down a given list of words and describing a visual scene in own words. Instead of predicting a fixed set of affective states, our approach can capture different affective regions which can be defined according to the researchers need. Our approach is not restricted to copying predefined sentences and figures but works with arbitrary handwriting and drawing. To our knowledge, this is the first work to leverage stylus data in order to predict the affective state of a student during math solving.

3. METHOD

We present a classification pipeline that automatically predicts affective states based on low-cost and mobile bio-sensor and stylus devices. Our pipeline assumes that we have access to reports on affective states of users based on the circumplex model of affects [29]. The circumplex model is a two-dimensional model representing affective states in terms of valence and arousal. The classification task then amounts to classifying regions within this space using a combination of signals from bio-sensor and stylus devices. For this purpose, we build a generic affective predictor (Figure 1). Recorded stylus and bio-sensor data are preprocessed and the relevant features are extracted to train a classification model for the specific affective regions. We design our predictor to work unobtrusively in the background of any ITS.

3.1 Input Signals

During the task solving process bio-sensor and stylus data are recorded.

Electrodermal activity (EDA). EDA is an indicator of the emotional state of a person reflected by the variation in the electrical characteristics of the skin as a result of sweating [2]. EDA is quantified by measuring the amount of current flowing between electrodes attached to the skin. Changes in affective states can lead to subtle variations in the level of sweat that can be detected as the changes in the current. Typically, the EDA signal is decomposed into tonic (low frequency) and phasic (high frequency) components.

Interbeat Intervals (IBIs). IBIs are the time intervals between consecutive heartbeats in normal heart function. This natural variation is also known as heart rate variability (HRV). The heart rate (HR) can be easily computed as the inverse of the IBI averaged over a certain time window.

Skin Temperature (ST). ST measures the thermal response of human skin. Vasoconstriction (e.g., provoked by

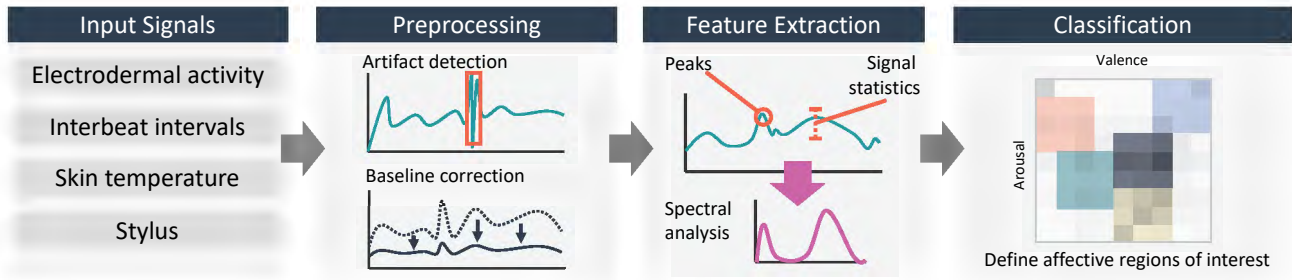


Figure 1: The classification pipeline. Stylus and bio-sensor data are gathered during task solving processes. After preprocessing the signals, features are extracted and used to classify the affective regions of interest.

an affective state) can result in an increase in blood flow and a consequent increase in ST [19].

Stylus. Tablet devices often come equipped with stylus pens as accessories that can provide precise and pressure-sensitive input. Stylus data consists of the applied pressure during writing and the pixel positions of the written text. From these measurements, handwriting characteristics related to time and ductus can be calculated.

3.2 Preprocessing of Signals

During preprocessing, the raw input signals are filtered in order to detect artifacts from movement and muscle contraction. The signals are also corrected for differences between individuals using baseline recordings for each individual.

Artifact detection. We follow the procedure outlined by Greco and colleagues [15] to decompose the EDA into tonic, phasic and an additive white Gaussian noise component with a convex optimization approach that accounts for signal filtering and detrending. For IBIs, detrending is not necessary in the preprocessing [36], and we use the criterion beat difference for artifact detection [16].

Baseline correction. Similar to previous work [30, 17], we collect baseline data for all sensors in order to account for individual differences in stylus and bio-sensor signals related to writing habit, ambient temperature and dryness of the skin. Baseline data is collected while individuals remain in a relaxed state (e.g., watching a nature video). We search for the minimum value of each bio-sensor signal during the relaxation phase over a 10 seconds window using a sliding window approach to be robust against outliers. Due to possible signal lags, we search the minimum for each signal separately. We then normalize the bio-sensor data by subtracting the feature values calculated over the corresponding 10 seconds interval of the baseline from the actual feature values computed during task solving. Stylus data is normalized by subtracting a baseline for all features computed over handwriting of an English sentence.

3.3 Feature Extraction

In the proposed pipeline, we extract several different feature types from the stylus and bio-sensor signals. Where appropriate, we compute basic statistics for these features types including the mean, standard deviation (SD), minimum and maximum and the linear trend (slope of a fitted linear regression line). A summary of all extracted features is presented in Table 1.

EDA. For EDA, we decompose the signal into phasic and tonic components and calculate standard statistics (i.e., mean, SD, min, max, slope). For the phasic component, we also calculate the area under the curve (AUC) [3] and the number of peaks using zero-crossings of the smoothed gradients of the signal [19]. Based on the extracted peaks, we further compute amplitude statistics (i.e., mean, min, max) [38].

IBI. From the IBI recordings, we extract temporal and frequency features. In the temporal domain, we calculate the percentage of successive IBIs that differ by more than 50 milliseconds (pNN50) and 20 milliseconds (pNN20) as well as the SD and root mean square of successive differences between adjacent IBIs (SDSD and RMSSD) [34, 25]. For the frequency domain, it is well known that the distribution of spectral power gives an indication of physiological activation [3]. Therefore, we extract a feature related to the high frequency (HF) band of 0.15-0.40 Hz by a Fast Fourier transform of the cubic spline interpolated signal [34, 25]. Based on the IBIs, we compute the heart rate for which we extract several standard statistics (i.e., mean, SD, min, max, slope).

ST. We extract several statistics (i.e., mean, SD, min, max, slope) from the temperature signal [38, 35].

Stylus. From the stylus data, we derive features related to the pressure applied by the pen as well as timing and location information. Previous research has successfully employed these features to predict affective states [24, 12]. From the pressure data, we compute standard statistics (mean, SD, max, min) per stroke and average these over an entire task. Additionally, over each task we compute the slope of a linear regression fit to the pressure values and the statistical skewness of the pressure distribution. We also compute standard statistics (i.e., mean, SD, max, min, slope) of the speed and acceleration of the strokes. For the handwriting data, we discriminate between the actual writing process and the think time while completing the task [24]. During writing there are always small time gaps between strokes which cannot be attributed to thinking but belong to the writing process itself. Because writing patterns are different for every user, we infer an individual threshold for each user to distinguish if the time between two strokes belongs to thinking or to the actual writing process. We chose this threshold as the 80 % cut-off value of the distribution of the time between the strokes over the stylus baseline (cropping the right tail of the distribution). Based on this threshold, we derive a feature measuring the percentage of writing (i.e., the time spent in the writing process). Additionally, we compute the

statistics (i.e., mean, SD, max, min) on the speed between consecutive strokes having time differences below threshold (writing process) and on the distance between strokes having time differences above threshold (thinking).

Table 1: Extracted bio-sensor and stylus features. For each signal, the features are sorted according to their importance (based on our experiments). The 10 most predictive features are highlighted in bold. SD refers to the standard deviation.

Signals	Features
EDA	Phasic AUC, Phasic Mean, Tonic SD, Tonic Max, Tonic Mean, Tonic Min, Phasic SD, # Phasic Peaks, Tonic Slope, Max Phasic Peak Amplitude, Min Phasic Peak Amplitude, Phasic Slope, Mean Phasic Peak Amplitude
Heart	IBI SDSD, IBI RMSSD, IBI SD, IBI pNN20, HR Mean, IBI High Frequency, IBI pNN50, IBI Mean, HR Min, HR Max, HR SD, HR Slope
Temperature	Max, Mean, Min, Slope, SD
Stylus	#Strokes/Mean Speed, Mean Distance between Strokes, Max Distance between Strokes, SD Distance between Strokes, Mean Pressure, Max Pressure, Mean Stroke Acceleration, Max Stroke Acceleration, Max Stroke Speed, Max Speed between Strokes, Mean Speed between Strokes, SD Speed between Strokes, SD Stroke Speed, SD Stroke Acceleration Excluded ¹ : %Writing, {SD, Slope, Skewness} Pressure, {Mean, Min, Slope} Stroke Speed, {Min, Slope} Stroke Acceleration, Min Speed between Strokes, Min Distance between Strokes, #Strokes/Minute

¹ Excluded due to our experimental setup (see Section 5.1)

3.4 Classification

To train our classification algorithms ground truth is built by defining arbitrary non-overlapping regions of interest in the two-dimensional valence and arousal space based on the affective labels which can be gathered, for example, through self-reports or expert labelers. We then use a classification model to predict the affective region an individual is likely to be in during task solving based on the recorded bio-sensor and stylus data. Before applying the classification algorithm, we standardize all features to have zero mean and unit variance. We propose the usage of four different classifiers (i.e., Random Forest, Support Vector Machine, k-Nearest Neighbors and Gaussian Naive Bayes). We select these classifiers because they are among the most widely used in machine learning and have shown to provide good results on bio-sensor and stylus data [37, 24, 13]. All models are evaluated using leave-one-user-out cross-validation which ensures that data from the same user is not in the testing and training set at the same time. Hyperparameter optimization is performed using nested cross-validation and randomized search.

4. EXPERIMENT

We conducted a controlled lab experiment with 88 participants in order to test our pipeline. In the experiment, we recorded bio-sensor and stylus data while participants solved

approximately 40 math tasks chosen to trigger different affective states. The math tasks were chosen because they are an integral part of the educational curriculum. However, instead of relying on a math based ITS, we have designed specific math tasks to increase the probability of evoking a wider range of affective states.

4.1 Experimental Setup

Participants. We recruited 88 participants (45 female) between ages of 18 and 29 (mean = 22.1, SD = 2.0) from 10 different engineering and natural science departments of the second and third year of the Bachelor program of an university. We excluded participants suffering from cardiovascular pathologies, smokers, and participants suffering from evident mental pathologies (score > 4 in the Patient Health Questionnaire [22]). In order to control for external factors, we kept the humidity and room temperature at an average of 21.7 °C (SD = 0.59 °C) and 32.6 % (SD = 5.3 %), respectively. Figure 2 presents the experimental setup.

Sensors. We measured EDA and wrist acceleration using a Shimmer GSR+ device. To test the accuracy of the device, we compared its measurements with a state of the art ADInstruments PowerLab 8/35 device (connected through the ADInstruments FE116 GSRAmp signal amplifier) over a 23 minute recording of an user watching a nature video and picture stimuli. Results revealed a strong and significant cross-correlation value of 0.96 (p -value < 10^{-100}) between the two signals. These results suggest that the smaller, mobile and more affordable Shimmer GSR+ device may be sufficient to detect changes in affective states. During the experiment, the Shimmer GSR+ device was worn on the non-dominant hand with the electrodes placed at the proximal phalanx of the index and middle finger [7]. Data was recorded at a sampling rate of 100 Hz. As part of the Shimmer GSR+ setup, we also attached an optical pulse sensor providing a photoplethysmogram signal on the ring finger. However, photoplethysmogram data was of poor quality and consequently discarded from analysis. Prior to electrode attachment, we asked participants to wash their hands with lukewarm water [5]. Heart activity was measured using a Polar H10 chest belt. The Polar H10 belt provides IBIs and post-processed heart rate data by monitoring electrical changes on the surface of the skin. A predecessor of this device (Polar H7) was shown to provide accurate data when compared to an expensive lab device (Cosmed Quark T12x system) [28]. We recorded the skin temperature using the infrared thermopile sensor of the Empatica E4 device (sampling rate = 4 Hz; resolution = 0.02 °C). Since the sensor was attached to the dominant hand (used for writing during the tasks), other signals that the wristband can provide (EDA and blood volume pulse) were heavily affected by motion artifacts and discarded from the analyses.

During the experiment, participants interacted with a Huawei MediaPad M2 10.0 running Android 5.1 to solve the different math tasks. All interactions with the tablet were conducted with a Wacom Bamboo Ink stylus at an average sampling rate of 250 Hz (SD = 25 Hz) and with 2048 levels of pressure sensitivity. The signals from the bio-sensor devices were streamed to the tablet using the Bluetooth Low Energy protocol. We also recorded the behaviour of participants with the front camera of the tablet and a GoPro HERO3 camera.

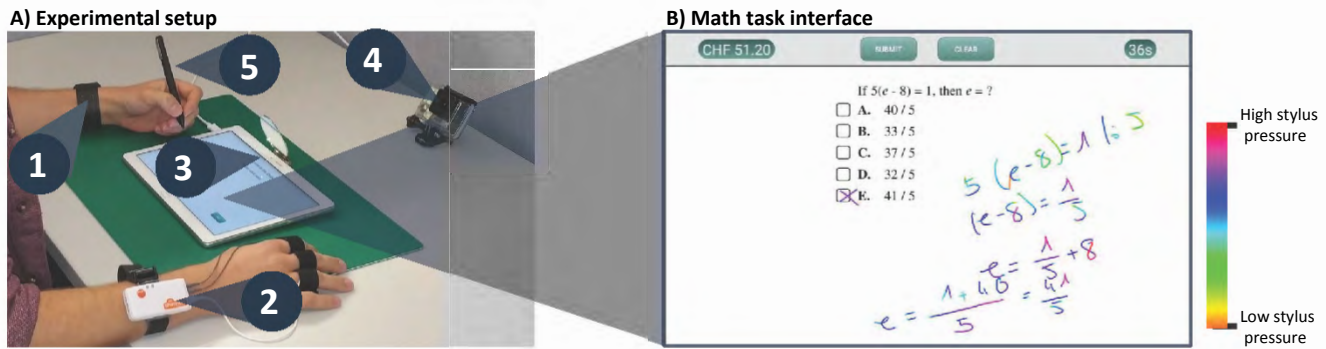


Figure 2: A participant completing the math tasks. A) During each session data is recorded from different devices. (1) An Empatica E4 recording skin temperature on the dominant hand. (2) A Shimmer GSR+ measuring skin conductance and wrist acceleration on the non-dominant hand. Participant behaviour was recorded by (3) the tablet front cam and (4) a GoPro HERO3. All interactions with the tablet were conducted with a stylus (5). Participants also wore a Polar H10 chest belt (not visible in the image) for recording heart activity. B) The task interface allows participants to write solution paths directly onto the screen (the stylus pressure is color-coded for visualization purposes only).

4.2 Experimental Procedure

For measuring the affective states of the participants we have used the self-assessment manikin (SAM) [6]. The SAM provides valence and arousal labels on a scale from 1 (most negative, lowest arousal) to 9 (most positive, highest arousal). For triggering the affective states we have used math tasks and pictures from the International Affective Picture System (IAPS) [23]. The IAPS is a database of 1182 pictures typically used in emotion research and has been standardized in terms of valence and arousal based on SAM ratings. We used the IAPS to investigate whether the affective model for the math tasks generalized to passive tasks, such as watching pictures (Section 5.6). As such, the set of IAPS pictures presented to the participants was sampled to cover similar affective regions as those expected to be evoked by the different math tasks.

An overview of the study procedure is presented in Figure 3A. The experiment lasted an average of 90 minutes for each participant. Upon arriving at the lab, participants completed a demographics questionnaire and were given an oral overview of the procedure. This included an explanation of the SAM questionnaire based on 4 example pictures from the IAPS presented on paper. Next, participants started working independently on the tablet by first watching a 7 minute nature video (bio-sensor baseline), followed by the stylus baseline that consisted of writing an English sentence with the stylus. Participants were then presented with 40 pictures from the IAPS in random order. Each picture was shown for 10 seconds and was directly followed by the SAM rating (valence and arousal) and a 10 second fixation cross. In total, we collected 3400 ratings from all participants. After rating the IAPS pictures, participants were asked to watch the nature video one more time before completing the math tasks. Before finishing the experiment, participants completed a paper questionnaire about their overall mood, comfort level while wearing the sensors, nervousness and sweating level.

4.3 Experimental Tasks

To trigger different affective states, we have created three different math task conditions by varying the difficulty level,

available time for completion and monetary reward of the task. These types of manipulations were shown to be effective at eliciting different affective states in reading comprehension [4] and math tasks [32].

Task design. The math tasks were taken from an ACT data set [1] that provided difficulty ratings from 0.12 (most difficult) to 0.96 (simplest). We conducted a pilot study (exact same conditions, 11 participants) to get an indication of the time needed to solve the different tasks. Based on this timing information and the tasks from the ACT data set we generated the following three conditions.

1) *Repetitive condition.* For the *repetitive condition* we created random variants (by substituting the numerical values in the task) of two easy tasks from the ACT data set (difficulty = 0.76 and 0.83). The time available to solve each task was set between 60 and 75 seconds at random. This provided participants with more than sufficient time to come up with a solution for each task. Correctly solving a task in the *repetitive condition* granted only a minor monetary reward (+CHF 0.2) and a minor penalty (-CHF 0.2) for incorrect solutions. The *repetitive condition* was designed to trigger emotions such as boredom and fatigue.

2) *Challenge condition.* For the *challenge condition* we selected math tasks from the ACT data set with medium difficulty (difficulty $\in [0.58, 0.69]$) and provided participants with a larger monetary reward (+CHF 2) for correct solutions and the same small penalty as the *repetitive condition* (-CHF 0.2) for incorrect solutions. Participants were provided with sufficient time to solve the tasks based on data from the pilot study (min = 53 seconds, max = 93 seconds). The *challenge condition* was designed to provide diversified tasks for a more engaging and interesting experience, while the larger monetary reward provided a bigger incentive (higher-stakes) for participants to perform well with relatively small penalty in case of mistakes.

3) *Overchallenge condition.* For the *overchallenge condition*, we selected the math tasks with high difficulty in the

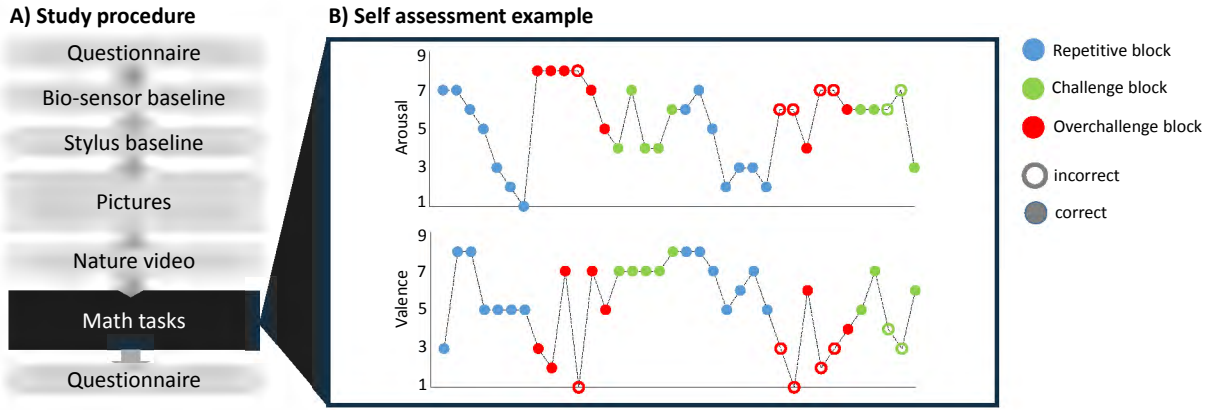


Figure 3: Overview over the different parts of the study. A) Overall experimental procedure. B) Changes in valence and arousal for one participant in relation to task type and answer.

ACT data set (difficulty $\in [0.25, 0.53]$). Participants received small monetary rewards for correct solutions (+CHF 0.2) and a large penalty (-CHF 2) for incorrect solutions. The time to solve each task was set to be insufficient for most participants based on data from the pilot study (min = 25 seconds, max = 51 seconds). The *overchallenge condition* was designed to provide a frustrating and annoying experience to participants.

The math tasks were presented in six blocks (2 in each condition) each containing a different number of tasks (*repetitive condition* 13 tasks, *challenge condition* 5 tasks, *overchallenge condition* 6 tasks). A similar block design for math tasks was already applied in previous work [32]. Moreover, we believe that a sequence of tasks is necessary to trigger an affective state. The first 3 blocks presented were randomly sampled. However, the succeeding 3 blocks were fixed to the same order as the first 3 blocks (but contained different tasks). In addition, the maximum time for each block was limited to 5 minutes to ensure that the math part of the experiment does not go over 30 minutes. After each block, a fixation cross was shown for 30 seconds to reduce potential carry-over effects of affective states. At the end of each math task, participants were asked to fill in the 9-point SAM scale to report their current valence and arousal level (in total, we have collected 3026 ratings from the participants). Figure 3B depicts the changes in the valence and arousal ratings for one participant in relation to the block type and task answer (correct vs. incorrect). We see that for the repetitive tasks, valence and arousal are decreasing over time leading to a shift towards boredom. Additionally, for incorrectly solved tasks, valence drops and arousal tends to increase. After the repetitive blocks we see a decrease in valence and an immediate steep increase in arousal that may be attributed to the increase in difficulty from the repetitive block to the overchallenge block. On average participants finished with CHF 44.3 (min = CHF 22.2, max = CHF 62.8). At the end of the experiment, each participant was compensated with a minimum of CHF 40.

Math task interface. Participants were asked to provide a solution path for every task anywhere on the screen and then to select their answers from 5 multiple-choice alternatives (see Figure 2B). Participants received immediate feedback

on whether their answer was correct. A timer located on the top right corner of the interface informed participants about the time left to respond and started to blink when less than 10 seconds remained. When the time was up and the participant did not submit a solution, the answer was considered wrong. The cumulative amount of money earned was displayed on the top left of the interface.

5. RESULTS

We compared different versions of our classification pipeline using only a subset of the sensors with a focus on the difference between stylus and bio-sensors. All results are based on Random Forest (using 500 trees, balanced class weights and hyperparameter optimization using randomized search with 100 iterations) given that this was the best performing classifier. In order to measure the performance of our classifiers, we have used accuracy (chance level = $1/\#$ classes) and micro-averaged area under curve (AUC) of the receiver operating characteristic (ROC) curve (chance level = 0.5), which aggregates the contributions of all classes to compute the average metric. Because both metrics are affected by class imbalance, we have also considered the macro-averaged AUC (chance level = 0.5) which is the average of the class-wise AUCs giving each class the same weight. To derive the SD for each metric, we employed an additional 10-fold cross-validation.

5.1 Study Validation

Our study was designed to trigger affective states across the entire valence-arousal space. As a first step, we investigated if our study design worked by examining if the different parameters acted as intended. In our task design we varied task difficulty, monetary reward and the available time for task completion. We have performed a per task Kendall's tau correlation analysis between these 3 parameters and the arousal and valence ratings of the participants. For the task difficulty and the percentage of remaining time, we have found high correlations for both valence (-0.2 ; p -value $< 10^{-59}$ and 0.22 ; p -value $< 10^{-80}$) and arousal (0.27 ; p -value $< 10^{-102}$ and -0.27 ; p -value $< 10^{-117}$). Participants shifted towards frustration (decreasing valence and increasing arousal) with increasing task difficulty or with a reduction in the time remaining to complete the task. In-

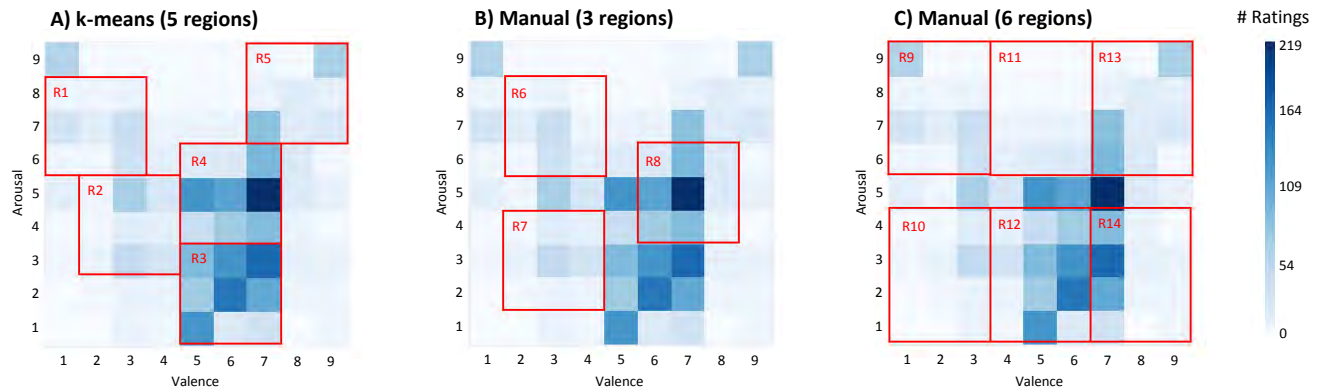


Figure 4: Heat maps showing the distribution of the participants' ratings on the math tasks. The red rectangles represent the different regions. A) 5 regions automatically chosen using k-means clustering. B) Three regions manually selected. C) Six regions manually selected.

terestingly, the effect size on valence and arousal is almost identical. In contrast, monetary reward appears to have a much larger effect on valence (0.47 ; $p\text{-value} < 10^{-295}$) than on arousal (-0.06 ; $p\text{-value} < 10^{-4}$). Altogether, it appears that our tasks worked as intended. Accounting for potential superficial correlations (e.g., task duration) is an important part of our study design. We found a significant Kendall's tau correlation between the task duration and the user ratings of 0.17 ($p\text{-value} < 10^{-48}$) and -0.11 ($p\text{-value} < 10^{-22}$) for arousal and valence, respectively. Because we have extracted the stylus features over the whole tasks, we have excluded all features having a significant Spearman correlation to the task duration (features greyed out in Table 1).

5.2 Data Analysis

Input Signals. Given that we detected a very low amount of artifacts across participants (EDA = 0.015% and IBI = 0.71%), we refrained from removing them from the analysis. Visual inspection of the ST recordings revealed a slow linear increase of the temperature over the course of a participant's session. This change in temperature may be due to the skin warming up under the wristband and independent of the affective state of the participants. We removed this linear trend from all measurements by subtracting the result of a linear least-squares fit to the signal. We did not observe any other artifacts for ST. The bio-sensor features listed in Table 1 have been computed using a window of 10 seconds since the minimum task duration was 10 seconds. For the stylus features, we have used an implicit window over the entire task. In addition, we have excluded all data points having at least one missing value.

Clustering of Ratings. Figure 4 presents the distribution of the participants' ratings in the valence-arousal space (dark and light blue refers to a high and low number of data points, respectively). A v-shape is visible with most ratings being made at a valence and arousal level of 7 and 5, corresponding to a positive medium intense state (e.g., interest). Several ratings were made at the extremes (top left and top right) of the valence-arousal space corresponding to states of distress and excitement that are associated with very good and very poor performance. To uncover the underlying clusters in the data, we have applied k-means

clustering in this two-dimensional valence and arousal space. Using the Bayesian information criterion, we found an optimal number of 5 clusters. We defined region boundaries (shown by the red rectangles in Figure 4A) as the arithmetically rounded value of the centroid of each cluster plus and minus the standard deviation of the participants' ratings in the corresponding cluster. We observed that the regions are all of equal size and cover the area of the v-shape. Based on Russell's [29] and Scherer's [33] categorization we identify the following regions, their sizes and corresponding affective states: Region R1 (213 data points; frustrated, annoyed), region R2 (284; bored, taken aback), region R3 (965; attentive, serious), region R4 (861; expectant, confident), region R5 (295; excited, triumphant). Together, it appears that the math task covered a broad range of affective states relevant for learning and that positive states (R3, R4, R5) dominate.

5.3 Classification Performance

Figure 5A and Table 2 present the predictive performance of the model based on the 5 defined regions. Using all sensors, the model achieved an accuracy of 65% (chance level = 20%). Here, the slightly lower value for the macro-averaged AUC (0.83) compared to the micro-averaged AUC (0.88) may be related to class imbalance. Figure 5C depicts the confusion matrix based on all sensors. The matrix shows that regions R1 and R2 are more difficult to predict than the other regions. This may be due to the lower number of data points collected for these regions. As expected, the larger the distance between the regions, the easier it is for the model to discriminate between them.

Feature Importance. Table 1 presents the 10 most important features (in bold). The features are sorted according to their relative importance which we computed using permutation feature importance (permuting each feature 100 times and measuring the mean decrease in micro-averaged AUC). We obtained the same relative feature importance ordering using the Gini importance measure. EDA and heart measures provided 3 out of the 10 most important features and stylus features contributed with 4 of the most important features. There were no ST features among the top ten features. Regarding the heart measures, the features related to IBIs were more important than HR features. An

interesting observation can be made for the stylus features. Features related to the distance between strokes appear to be more important than speed between stroke features indicating that the spread of writing attributed to thinking (i.e., how the writing space is covered) provides more information than the actual writing behaviour.

5.4 Sensor Comparison

Bio-Sensors. If we consider the individual sensors (Figure 5B), ST performs substantially worse (-0.11 AUC) compared to EDA (0.80 AUC) and heart rate measures (0.81 AUC). The combination of all the bio-sensors (Figure 5A) provides only marginal performance improvements ($+0.05$ AUC) compared to the individual sensors.

Stylus. Our most important finding is that the stylus performs equally well as the bio-sensors (Figure 5B), rendering the data from the bio-sensors redundant and unnecessary for the prediction of affective states. The performance of the stylus is only marginally inferior (-0.02 AUC) when compared to the combination of all bio-sensors. In contrast, the combination of the bio-sensors and the stylus achieves a slightly higher performance ($+0.02$ AUC) compared with the bio-sensors and stylus alone (Figure 5A). This might be an indication that they may contain complementary information, although the difference appears to be small.

5.5 Affective Region Analysis

In order to investigate the ability of our pipeline to predict different affective regions based on the recorded bio-sensor and stylus data we have defined two additional coverings of the valence and arousal space (Figure 4B and 4C). Based on Russell [29] and Scherer [33] we have manually defined specific regions associated with frustration (annoying; region R6, 185 data points), boredom (taken aback; region R7, 199) and interest (engaged concentration, flow; region R8, 720) as shown in Figure 4B. Being able to distinguish these 3 regions is important in education due to their impact on learning gain [9, 26, 10]. To cover the valence and arousal space evenly, we have manually defined the 6 regions shown in Figure 4C, dividing arousal in two and valence in 3 components (The number of data points from region R9 to R14 are 287, 154, 134, 852, 432 and 506). The results for both space partitionings are listed in Table 2 (note that chance level for the accuracy is 33 % for 3 regions and 16.66 % for 6 regions). The performance of the classification of 3 regions outperforms the one for 5 and 6 regions in terms of accuracy. On the other hand, when taking into account the AUC, there is no substantial difference in performance between the different coverings. This difference between accuracy and AUC stems from the fact that predicting only 3 regions is a much easier task than predicting 5 or 6 regions. This is in line with the finding that the accuracy for predicting 5 regions is slightly higher than for 6 regions. Nevertheless, we can conclude that we have seen that our approach is able to provide good results for 3 different coverings. Thus, we come to the conclusion that our pipeline is rather flexible being able to handle different regions in the valence-arousal space. Compared to previous work relying on fixed affective states, our approach has the advantage that the regions do not have to be pre-defined allowing for much more flexible use.

Table 2: Performance of Random Forest on the math data for different signals and regions. AUC_{micro} and AUC_{macro} represent micro-averaged and macro-averaged AUC, respectively. The chance level for accuracy is $1/\#$ regions and for AUC it is 0.5. The standard deviations are given in brackets.

Regions	Signals	AUC_{micro}	AUC_{macro}	Accuracy
k-means (5 Regions)	EDA	0.80 (0.02)	0.75 (0.03)	50 % (4 %)
	Heart	0.81 (0.01)	0.73 (0.01)	52 % (2 %)
	Temperature	0.69 (0.03)	0.59 (0.03)	37 % (4 %)
	Stylus	0.84 (0.01)	0.76 (0.02)	59 % (2 %)
	Bio-Sensors	0.86 (0.01)	0.81 (0.02)	60 % (2 %)
	Bio-Sensors & Stylus	0.88 (0.01)	0.83 (0.02)	64 % (2 %)
Manual (3 Regions)	EDA	0.81 (0.02)	0.69 (0.04)	66 % (2 %)
	Heart	0.79 (0.02)	0.66 (0.03)	62 % (3 %)
	Temperature	0.76 (0.01)	0.60 (0.04)	60 % (3 %)
	Stylus	0.83 (0.02)	0.72 (0.02)	67 % (3 %)
	Bio-Sensors	0.84 (0.01)	0.76 (0.03)	67 % (1 %)
	Bio-Sensors & Stylus	0.87 (0.01)	0.80 (0.02)	67 % (2 %)
Manual (6 Regions)	EDA	0.80 (0.02)	0.72 (0.03)	46 % (3 %)
	Heart	0.78 (0.01)	0.72 (0.02)	44 % (2 %)
	Temperature	0.70 (0.02)	0.61 (0.02)	35 % (3 %)
	Stylus	0.81 (0.01)	0.75 (0.02)	48 % (2 %)
	Bio-Sensors	0.85 (0.02)	0.80 (0.02)	57 % (4 %)
	Bio-Sensors & Stylus	0.87 (0.02)	0.83 (0.03)	61 % (3 %)

5.6 Model Transfer

In addition to the math tasks, we have also gathered bio-sensor data as well as valence and arousal ratings from the participants while they observed pictures from the IAPS. We have used this data to investigate our model's capacity to generalize to more passive tasks, such as looking at pictures. To predict the affective regions of interest, we applied our model trained on the bio-sensor data recorded during math task solving to data collected while participants viewed and rated the set of IAPS pictures. When we consider the 5 different regions (Figure 4A), the model's accuracy reaches 39 % (chance level = 20 %, $AUC_{micro} = 0.68$, $AUC_{macro} = 0.64$). When we train and evaluate a model directly on the picture data, we achieve a slightly better classification performance (accuracy = 42 %, $AUC_{micro} = 0.74$, $AUC_{macro} = 0.66$). There may be several reasons behind the suboptimal performance when predicting affective states during the picture task. These include sociocultural aspects when rating emotions based on pictures (e.g., rating how it is expected), old and low resolution pictures from the IAPS data set, media influence desensitizing participants to the content of the IAPS, and the fact that the math and picture domains are very different. Together these initial results indicate that building a general predictor of affective states might be possible, but further experiments are necessary.

6. CONCLUSION

In this paper we presented a generic pipeline for predicting affective regions of interest using bio-sensor and stylus data. We validated our pipeline for the case of math solving tasks and demonstrated that our pipeline can accurately predict various regions in the valence-arousal space (up to 0.88 AUC). In addition, we have compared different input signals with each other. The performance of the Shimmer GSR+ and Polar H10 have been on the same level (up to 0.81 AUC). Moreover, we found that the classification performance using only stylus data is comparable to the classification performance based on the bio-sensors. Taking into account the emerging digitization of education and the spread of tablets in schools and private households, these results

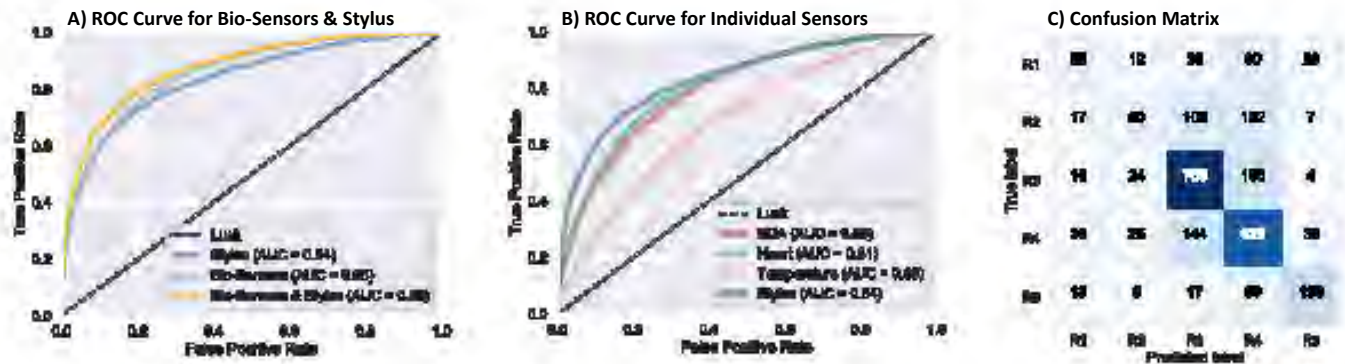


Figure 5: ROC curves and micro-averaged AUC scores for 5 regions chosen by k-means clustering for (A) the bio-sensors, stylus and the combination of bio sensors and stylus and (B) the individual bio-sensors & stylus. (C) The confusion matrix is computed by using the combination of bio-sensors and stylus.

make the stylus a preferred alternative to bio-sensors for measuring affective states in classrooms. Using bio-sensors in classroom settings can be cumbersome and costly as it requires the purchase and synchronization of several devices. In contrast, systems that depend on a stylus only are cheaper than systems relying on bio-sensor devices, and styluses often come bundled with mobile devices, such as tablets or smartphones. In addition to being cheaper and more ubiquitous, styluses are easier to setup (e.g., no attachment of electrodes, no motion artifacts) and less intrusive. Furthermore, stylus data is not only restricted to digital devices but can also be recorded using digital pens. Finally, we have demonstrated the possibility of a generalized model for predicting affective states by applying the model trained on the data from the math tasks (active part) to pictures from the IAPS (passive part) reaching a performance of 0.68 AUC.

There are some potential limitations to the approach presented here. First of all, the setup is restricted to a lab environment and the population of Bachelor students may limit generalization to students at other levels. We are optimistic that our approach also works outside a controlled setting and for a broader population. Participants reported that the setup was comfortable and that they could act in a natural way. In addition, we assume that given a proper baseline correction the signals are also predictive for a heterogeneous group of people. Another limitation is the restriction to math tasks. Similar to bio-sensor data, we believe that handwriting data carries affective information independent of the task. Thus, we expect our approach to work also in other domains involving handwriting, such as solving exercises for different school subjects and writing essays.

Future research from our lab will test and refine our pipeline for multiple domains. Potential refinements include using non-linear IBI features and frequency features for skin temperature. Additionally, an in depth analysis of handwriting that takes into account the slant of the handwriting could further improve the classification performance. Another interesting direction would be to make use of large existing bio-sensor databases for semi-supervised learning by using auto-encoders to infer an efficient feature embedding [20].

7. REFERENCES

- [1] ACT. The act technical manual, 2017.
- [2] M. Benedek and C. Kaernbach. A continuous measure of phasic electrodermal activity. *Journal of neuroscience methods*, 190(1):80–91, 2010.
- [3] A. Betella, R. Zucca, R. Cetnarski, A. Greco, A. Lanatà, D. Mazzei, A. Tognetti, X. D. Arsiwalla, P. Omedas, D. De Rossi, et al. Inference of human affective states from psychophysiological measurements extracted under ecologically valid conditions. *Frontiers in neuroscience*, 8:286, 2014.
- [4] N. Blanchard, R. Bixler, T. Joyce, and S. D’Mello. Automated physiological-based detection of mind wandering during learning. In *Proc. ITS*, pages 55–60. Springer, 2014.
- [5] W. Boucsein, D. C. Fowles, S. Grimnes, G. Ben-Shakhar, W. T. Roth, M. E. Dawson, and D. L. Fillion. Publication recommendations for electrodermal measurements. *Psychophysiology*, pages 1017–34, 2012.
- [6] M. M. Bradley and P. J. Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994.
- [7] R. A. Calvo, S. D’Mello, J. Gratch, and A. Kappas. *The Oxford handbook of affective computing*. Oxford Library of Psychology, 2015.
- [8] C. Conati and H. Maclaren. Modeling user affect from causes and effects. In *Proc UMAP*, pages 4–15. Springer, 2009.
- [9] M. Csikszentmihalyi. *Flow: The Psychology of Optimal Experience*. Harper Perennial, New York, NY, 2008.
- [10] R. S. J. d Baker, S. M. Gowda, M. Wixon, J. Kalka, A. Z. Wagner, A. Salvi, V. Aleven, G. W. Kusbit, J. Ocumpaugh, and L. Rossi. Towards sensor-free affect detection in cognitive tutor algebra. *International Educational Data Mining Society*, 2012.
- [11] M. Ez-Zaouia and E. Lavoué. Emoda: a tutor oriented multimodal and contextual emotional dashboard. In *Proc. LAK*, pages 429–438. ACM, 2017.
- [12] M. Fairhurst, M. Erbilek, and C. Li. Study of automatic prediction of emotion from handwriting samples. *IET Biometrics*, pages 90–97, 2015.

- [13] T. Fritz, A. Begel, S. C. Müller, S. Yigit-Elliott, and M. Züger. Using psycho-physiological measures to assess task difficulty in software development. In *Proceedings of the 36th International Conference on Software Engineering*, pages 402–413. ACM, 2014.
- [14] B. Grawemeyer, M. Mavrikis, W. Holmes, S. Gutierrez-Santos, M. Wiedmann, and N. Rummel. Affecting off-task behaviour: how affect-aware feedback can improve student learning. In *Proc. LAK*, pages 104–113. ACM, 2016.
- [15] A. Greco, G. Valenza, A. Lanata, E. P. Scilingo, and L. Citi. cvxeda: A convex optimization approach to electrodermal activity processing. *IEEE Trans. on Biomedical Engineering*, 63(4):797–804, 2016.
- [16] K. Hovsepian, M. al’Absi, E. Ertin, T. Kamarck, M. Nakajima, and S. Kumar. cstress: towards a gold standard for continuous stress assessment in the mobile environment. In *Proc. of the international joint conference on pervasive and ubiquitous computing*, pages 493–504. ACM, 2015.
- [17] I. Jraidi, M. Chaouachi, and C. Frasson. A hierarchical probabilistic framework for recognizing learners’ interaction experience trends and emotions. *Advances in Human-Computer Interaction*, 2014:6, 2014.
- [18] T. Käser, G.-M. Baschera, A. G. Busetto, S. Klingler, B. Solenthaler, J. M. Buhmann, and M. Gross. Towards a framework for modelling engagement dynamics in multiple learning domains. *International journal of artificial intelligence in education*, 22(1-2):59–83, 2013.
- [19] K. H. Kim, S. W. Bang, and S. R. Kim. Emotion recognition system using short-term monitoring of physiological signals. *Medical and biological engineering and computing*, 42(3):419–427, 2004.
- [20] S. Klingler, R. Wampfler, T. Käser, B. Solenthaler, and M. Gross. Efficient feature embeddings for student classification with variational auto-encoders. In *Proceedings of EDM*, 2017.
- [21] V. Kostyuk, M. V. Almeda, and R. S. Baker. Correlating affect and behavior in reasoning mind with state test achievement. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*, pages 26–30. ACM, 2018.
- [22] K. Kroenke, R. L. Spitzer, and J. B. W. Williams. The phq-9: validity of a brief depression severity measure. *Journal of general internal medicine*, 16(9):606–613, 2001.
- [23] P. J. Lang, M. M. Bradley, and B. N. Cuthbert. International affective picture system (IAPS): Affective ratings of pictures and instruction manual. Technical Report A-8, The Center for Research in Psychophysiology, University of Florida, Gainesville, FL, 2008.
- [24] L. Likforman-Sulem, A. Esposito, M. Faundez-Zanuy, S. Cléménçon, and G. Cordasco. Emothaw: A novel database for emotional state recognition from handwriting and drawing. *IEEE Transactions on Human-Machine Systems*, 47(2):273–284, 2017.
- [25] M. Malik, J. T. Bigger, A. J. Camm, R. E. Kleiger, A. Malliani, A. J. Moss, and P. J. Schwartz. Heart rate variability: Standards of measurement, physiological interpretation, and clinical use. *European heart journal*, 17(3):354–381, 1996.
- [26] M. Miserandino. Children who do well in school: Individual differences in perceived competence and autonomy in above-average children. *Journal of educational psychology*, 88(2):203, 1996.
- [27] Z. A. Pardos, R. S. J. D. Baker, M. O. C. Z. San Pedro, S. M. Gowda, and S. M. Gowda. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. In *Proc. LAK*, pages 117–124. ACM, 2013.
- [28] D. J. Plews, B. Scott, M. Altini, M. Wood, A. E. Kilding, and P. B. Laursen. Comparison of heart-rate-variability recording with smartphone photoplethysmography, polar h7 chest strap, and electrocardiography. *International journal of sports physiology and performance*, 12(10):1324–1328, 2017.
- [29] J. A. Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980.
- [30] S. Salmeron-Majadas, M. Arevalillo-Herráez, O. C. Santos, M. Saneiro, R. Cabestrero, P. Quirós, D. Arnau, and J. G. Boticario. Filtering of spontaneous and low intensity emotions in educational contexts. In *Proc. AIED*, pages 429–438. Springer, 2015.
- [31] S. Salmeron-Majadas, R. S. Baker, O. C. Santos, and J. G. Boticario. A machine learning approach to leverage individual keyboard and mouse interaction behavior from multiple users in real-world learning scenarios. *IEEE Access*, 6:39154–39179, 2018.
- [32] M. Saneiro, O. C. Santos, S. Salmeron-Majadas, and J. G. Boticario. Towards emotion detection in educational scenarios from facial expressions and body movements through multimodal approaches. *The Scientific World Journal*, 2014, 2014.
- [33] K. R. Scherer. What are emotions? and how can they be measured? *Social science information*, 44(4):695–729, 2005.
- [34] F. Shaffer and J. P. Ginsberg. An overview of heart rate variability metrics and norms. *Frontiers in public health*, 5:258, 2017.
- [35] Y. Shi, M. H. Nguyen, P. Blitz, B. French, S. Fisk, F. De la Torre, A. Smailagic, D. P. Siewiorek, M. al’Absi, E. Ertin, et al. Personalized stress detection from physiological measurements. In *International symposium on quality of life technology*, pages 28–29, 2010.
- [36] C.-S. Yoo and S.-H. Yi. Effects of detrending for analysis of heart rate variability and applications to the estimation of depth of anesthesia. *Journal of Korean Physical Society*, 44:561, 2004.
- [37] J. Zhou, K. Hang, S. Oviatt, K. Yu, and F. Chen. Combining empirical and machine learning techniques to predict math expertise using pen signal features. In *Proceedings of the 2014 ACM workshop on Multimodal Learning Analytics Workshop and Grand Challenge*, pages 29–36. ACM, 2014.
- [38] M. Züger and T. Fritz. Interruptibility of software developers and its prediction using psycho-physiological sensors. In *Proc. of the Conference on Human Factors in Computing Systems*, pages 2981–2990. ACM, 2015.

Active Learning for Student Affect Detection

Tsung-Yen Yang¹, Ryan S. Baker², Christoph Studer³, Neil Heffernan⁴, and Andrew S. Lan⁵

¹Princeton University, ²University of Pennsylvania, ³Cornell University,
⁴Worcester Polytechnic Institute, ⁵University of Massachusetts Amherst

ABSTRACT

“Sensor-free” detectors of student affect that use only student activity data and no physical or physiological sensors are cost-effective and have potential to be applied at large scale in real classrooms. These detectors are trained using student affect labels collected from human observers as they observe students learn within intelligent tutoring systems (ITSs) in real classrooms. Due to the inherent diversity of student activity and affect dynamics, observing the affective states of some students at certain times is likely to be more informative to the affect detectors than observing others. Therefore, a carefully-crafted observation schedule may lead to more meaningful observations and improved affect detectors. In this paper, we investigate whether active (machine) learning methods, a family of methods that adaptively select the next most informative observation, can improve the efficiency of the affect label collection process. We study several existing active learning methods and also propose a new method that is ideally suited for the problem setting in affect detection. We conduct a series of experiments using a real-world student affect dataset collected in real classrooms deploying the ASSISTments ITS. Results show that some active learning methods can lead to high-quality affect detectors using only a small number of highly informative observations. We also discuss how to deploy active learning methods in real classrooms to improve the affect label collection process and thus sensor-free affect detectors.

Keywords

Active learning, L-MMSE estimation, student affect detection

1. INTRODUCTION

Intelligent tutoring systems (ITSs) have gradually seen more and more deployment over the years in real classrooms all over the world. Recently, large-scale randomized controlled trials have shown that they can lead to improved student learning outcomes [30] and affect [19]. However, even the

state-of-the-art ITSs cannot interact with students the way human instructors can. For example, in real classrooms, instructors can detect a student’s knowledge and affective states by observing their activity and behavior and then adjust their teaching strategy by changing the difficulty of practice questions or addressing negative affect [2, 23]. In particular, keeping students in positive affective states (e.g., engaged) is crucial since their affective states are found to be highly predictive of many metrics of academic performance and success, including test scores [28] and college enrollment [29]. Consequently, there exist many works on designing interventions [1, 10] to address negative affect. Examples of such interventions include selecting appropriate textual dialogues to help students engage [12], using an embodied agent to mirror and empathize with confused students [6], and providing motivational message to frustrated students [19].

1.1 Student Affect Detection

Many existing student affect detection methods employ physical and physiological sensors that make frequent observations of students when they are learning. Despite their effectiveness, these detectors are impractical for large-scale deployment in real classrooms due to cost and privacy constraints [16, 35]. On the other hand, there exists a family of “sensor-free” detectors, which uses only student activity data as they learn within ITSs to detect affect [4, 32, 37]. These detectors use machine learning-based classifiers to predict student affective states from a set of activity features [5]. These sensor-free detectors are more feasible for large-scale deployment than those sensor-dependent ones for two reasons. First, they are cost-effective since once constructed, they can operate in fully-automated fashion and can easily be integrated into ITSs and deployed at large scale. Second, they are more privacy-aware since activity data can be more effectively anonymized than data obtained from other sensors, e.g., video recordings of the students’ facial expressions.

Although sensor-free affect detectors are highly automated, the student affect state label collection process remains labor-intensive. The process for collecting these labels typically consists of human observers (including trained coders and/or the teacher) making observations of students in real classrooms and encode their affect into a collection of states. For example, in the Baker Rodrigo Ocumpaugh monitoring protocol (BROMP) for affect observation and coding, there are four affective states: boredom, confusion, engaged concentration, and frustration. The process typically proceeds in round-robin fashion, i.e., the human observer alternates

Tsung-Yen Yang, Christoph Studer, Ryan S. Baker, Neil T. Heffernan and Andrew Lan "Active Learning for Student Affect Detection" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 208 - 217

among students and observe one student in each observation interval according to a pre-defined, ad hoc schedule.

However, this data collection process is insufficient since the typical round-robin schedule cannot make full use of the limited time human experts have to make observations. The reason is that, due to the inherent diversity in student activity and affect, the affect states of some students during some observation intervals are more informative to the classifiers than those in other cases; A non-adaptive, ad-hoc observation schedule leads to a lot of missed opportunities to observe these more informative cases. Therefore, it is desirable to develop methods that adaptively select the most informative students to observe in each observation interval and recommend them to human observers. These adaptive methods can potentially lead to the collection of higher-quality data for the affect detector to train on without requiring additional human effort, which will ultimately improve affect detection.

1.2 Active Learning

Active learning refers to a family of machine learning methods that adaptively select the next most “informative” observation to a classifier [34]. These methods are designed for applications where one has access to abundant unlabeled data but can only selectively label a small portion of it. In this setting, there is a need to select data instances whose labels, once obtained, result in the largest improvement in classification quality. There exist numerous active learning methods with different metrics of informativeness; these methods have been found to be effective at reducing the amount of labeled data needed when combining with many classifiers including logistic regression [38], support vector machines [15], and deep convolutional neural networks [33]. See Section 3.1 for a more formal introduction to active learning.

Existing active learning methods are not always successful in practice; in some settings, no active learning methods can outperform the simple baseline approach of randomly selecting data instances to label [38]. One such setting is the “cold-start” setting, when one does not yet have access to a sufficient amount of data to build a good classifier. In this setting, the estimate of informativeness can be highly inaccurate. In affect detection, since there are typically hundreds of features used to summarize student activity in ITSs [5], a classifier needs a significant number of labels to reach reasonable quality. Therefore, the effectiveness of existing active learning methods will be limited in the initial part of the student affect label collection process. Another such setting is when the data is highly noisy; in this case, it is hard to identify informative observations. In affective detection, the affective state labels provided by human observers are highly subjective and thus noisy; the labels provided by different human experts may differ [27] significantly. Therefore, the effectiveness of existing active learning methods in affect detection will be limited by the noisiness of the data. Therefore, it is desirable to develop new active learning methods that are robust to small and noisy data.

1.3 Contributions

In this paper, we investigate whether active learning can be used to improve the efficiency and effectiveness of student

affective state label collection. We conduct a preliminary study using several classic active learning methods on an existing real-world student affect dataset collected from ASSISTments¹, a widely-used ITS. Motivated by the limitations of existing active learning methods when the data is small and noisy, we also propose a new active learning method that can excel in this setting. Our new active learning method leverages the recently proposed linear minimum mean squared error (L-MMSE) estimation framework [21, 22] to evaluate observation informativeness. This framework provides an exact, closed-form, and nonasymptotic analysis of the parameter estimation error for binary regression and is shown to be highly effective when data is small and/or noisy. Experimental results show that some active learning methods, especially our L-MMSE-based method, can reduce the number of labels needed to build high-quality, sensor-free affect detectors. We also discuss how to use active learning to improve data collection efficiency in real-world affect detection and possibly other quantitative field observation (QFO) tasks by building an interactive system that suggests human observers to make certain observations.

We emphasize that the purpose of the current work is *not* to improve affect detectors but rather to investigate whether one can collect better data to train them. Therefore, we resort to a simple logistic regression-based affect detector since it can be integrated with all existing active learning methods. More complicated, state-of-the-art deep learning-based detectors cannot be integrated with many active learning methods and thus do not offer us a complete view of active learning in affect detection.

2. RELATED WORK

ASSISTments is a free web-based platform that provides immediate feedback, on-demand hints, and scaffolding support to the many students who use it in classrooms and for daily homework [14]. The system has been used by hundreds of thousands of students and thousands of teachers, and has been found to be effective in improving learning outcomes and closing achievement gaps in a large-scale randomized controlled trial [30].

A significant amount of research has been conducted on the detection of student affect by aligning ASSISTments data to student affect labels collected in real classrooms using BROMP [27]. BROMP allows human observers to label a student in four often-studied affective states: engaged concentration [9], frustration [20], boredom [25], and confusion [8]. Initially, sensor-free affect detectors in ASSISTments leveraged a number of rule-based and statistics-based models; these models achieved performance substantially above chance, for new students from rural, suburban, and urban populations [4]. Later, the work in [36] improved upon these initial affect detectors by incorporating additional features on skills/knowledge components as well as statistics across the entire class. Most recently, the work in [5] applied deep learning methods to affect detection and produced a significant increase in detection accuracy. The key in that work is to use recurrent neural networks (RNNs), including its two popular variants in long short-term memory (LSTM) networks and gated recurrent unit (GRU) networks [13], to

¹<https://www.assistments.org/>

capture students' changing affect over time.

3. ACTIVE LEARNING

In this section, we will first review active learning and briefly describe how it can be used to improve the efficiency in QFOs. We will then review the L-MMSE estimation framework and introduce our new, L-MMSE-based active learning method.

3.1 Background on Active Learning

Supervised learning refers to a class of machine learning approaches where the task is to learn a function (usually, a classifier) that captures the relation between input-output (feature-label) pairs. The typical setup in supervised learning is that one observes all features and labels and can use them to train the classifier. Active learning, on the other hand, deals with the setting where one has control of the data label observation process; in this case, one has access to the feature values of all feature-label pairs but can select which one gets labeled next. Naturally, the most effective strategy is to train the classifier on observed labels and select the next label that is the most "informative" to the current classifier to observe [34]. There exist numerous active learning methods with different metrics of informativeness, e.g., entropy (or observation uncertainty) [24], expected error reduction [31], expected variance reduction [40], model change [7], etc. The goal of active learning is to only observe labels that are highly informative in order to learn the function more efficiently.

Concretely, we denote the functional relation between the features and labels as

$$\mathbf{y} \sim f_{\mathbf{x}}(\mathbf{D}),$$

where $\mathbf{y} \in \mathcal{A}^N$ is the vector of labels that contains a total of N observations. \mathcal{A} denotes the set of labels. $\mathbf{D} \in \mathbb{R}^{N \times P}$ denotes the matrix containing all feature values corresponding to each label. The column vectors corresponding to the rows of \mathbf{D} , i.e., the feature values of each observation, are denoted as \mathbf{d}_i , $i \in \{1, \dots, N\}$. Correspondingly, each element in the label vector is denoted as y_i , $i \in \{1, \dots, N\}$. $f_{\mathbf{x}}(\cdot)$ denotes the function that maps each input feature vector \mathbf{d}_i to each label y_i ; \mathbf{x} denotes the vector containing all parameters of the function. In regression problems, \mathbf{x} corresponds to the regression coefficient vector, while in neural networks, \mathbf{x} corresponds to the collection of all weights and biases that characterize the connections between hidden units.

The iterative process of active learning proceeds as follows. Suppose that one now has a set of $t-1$ observations, with $t \in \{1, 2, \dots, N\}$ and wants to select the next, t -th observation. Let \mathcal{O}_{t-1} and \mathcal{U}_{t-1} denote the *sets* (which contain indices) of all feature-label pairs (referred to as datasets) where the labels are observed and unobserved, respectively, and let $\hat{\mathbf{x}}_{t-1}$ denote the current estimate of the function parameters (trained on the subset of feature values and labels $\mathbf{D}_{\mathcal{O}_{t-1}}$ and $\mathbf{y}_{\mathcal{O}_{t-1}}$). Active learning methods then select the next observation i_t as

$$i_t = \operatorname{argmax}_{i \in \mathcal{U}_{t-1}} I(\mathbf{d}_i, \hat{\mathbf{x}}_{t-1}),$$

where $I(\cdot, \cdot)$ denotes a metric of how informative an observation i is to the current function. As an example, the simplest yet often most effective existing active learning method, uncertainty sampling, simply uses the entropy [13] as the metric

of informativeness:

$$I(\mathbf{d}_i, \hat{\mathbf{x}}_{t-1}) = - \sum_{a \in \mathcal{A}} p(y_i = a) \log p(y_i = a),$$

where $p(y_i = a) = f_{\hat{\mathbf{x}}_{t-1}}(\mathbf{d}_i)$ denotes the probability of a new observation with feature values \mathbf{d}_i taking on label a given the current function estimate parameterized by $\hat{\mathbf{x}}_{t-1}$. In other words, uncertainty sampling simply selects the next observation whose label the current classifier is the least certain of. After selecting the next observation, the classifier is re-trained using an updated observed dataset $\mathcal{O}_t = \mathcal{O}_{t-1} \cup \{i_t\}$. Then, in the next iteration of the active learning process, the next observation i_{t+1} is selected from an updated unobserved dataset \mathcal{U}_t .

In typical active learning settings, since one has access to all feature values, the unobserved dataset is simply updated by excluding the selected observation as $\mathcal{U}_t = \mathcal{U}_{t-1} \setminus i_t$. However, we emphasize that under real-world QFO settings, the set of unobserved observations can change entirely. For example, in the typical BROMP coding process, a human coder observes the affective state of one student in each observation interval (typically 20 seconds); therefore, in the next iteration, the set of unobserved dataset (which contains feature values that summarize student activity during the next observation interval) might change entirely.

3.2 Background on L-MMSE Estimation

The L-MMSE estimation framework put forward in [21, 22] enables the design of new estimators for a wide range of nonlinear classification and regression problems. It also offers a closed-form, exact, and nonasymptotic analysis of the estimation error for nonlinear problems, which is typically impossible to obtain. The key insight to the L-MMSE estimation framework is that even for nonlinear problems, well-crafted linear estimators that take the nonlinearity into account can achieve comparable performance to nonlinear estimators that are computationally extensive and hard to analyze. Therefore, it is an advanced estimation technique and shall not be confused with basic linear estimation methods like least squares.

In [22], the L-MMSE estimation framework is applied to binary (especially probit) regression, which is given by

$$\mathbf{y} = \operatorname{sign}(\mathbf{D}\mathbf{x} + \mathbf{w}),$$

where $y_i \in \{-1, +1\}$ denotes the binary-valued label for feature-value pair i . The vector $\mathbf{w} \in \mathbb{R}^N$ denotes a noise vector with i.i.d. standard normal random entries. Putting a zero-mean multivariate normal prior with covariance matrix $\mathbf{C}_{\mathbf{x}}$ on \mathbf{x} as $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_{\mathbf{x}})$, the L-MMSE estimator finds the best estimator of \mathbf{x} that is *linear* in the observation vector \mathbf{y} , i.e.,

$$\hat{\mathbf{x}} = \mathbf{W}\mathbf{y},$$

where \mathbf{W} is a suitably-chosen estimation matrix that achieves the minimum mean-squared error (MSE) defined as

$$\text{MSE} = \mathbb{E}_{\mathbf{x}, \mathbf{w}} [\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2].$$

For probit regression, a variant of binary regression, the L-MMSE estimator has a closed-form expression, given by $\mathbf{W} = \mathbf{E}^T \mathbf{C}_{\mathbf{y}}^{-1}$, with its corresponding MSE given by

$$\text{MSE} = \text{tr}(\mathbf{C}_x - \mathbf{E}^T \mathbf{C}_y^{-1} \mathbf{E}), \quad (1)$$

where

$$\begin{aligned} \mathbf{E} &= \left(\frac{2}{\pi}\right)^{1/2} \text{diag}(\text{diag}(\mathbf{C}_z)^{-1/2}) \mathbf{D} \mathbf{C}_x, \\ \mathbf{C}_y &= \frac{2}{\pi} \sin^{-1} \left(\text{diag}(\text{diag}(\mathbf{C}_z)^{-1/2}) \mathbf{C}_z \right. \\ &\quad \left. \times \text{diag}(\text{diag}(\mathbf{C}_z)^{-1/2}) \right), \\ \mathbf{C}_z &= \mathbf{D} \mathbf{C}_x \mathbf{D}^T + \mathbf{I}. \end{aligned}$$

We note that the MSE (and also the matrix \mathbf{W}) depends only on the matrix \mathbf{D} and not the label vector \mathbf{y} .

3.3 L-MMSE-based Active Learning

Results in [21, 22] have shown that the L-MMSE estimator for binary regression performs on-par with state-of-the-art, sophisticated estimators, e.g., those that require using tools in convex optimization and Markov chain Monte Carlo techniques, while having much lower computational complexity. More importantly, the L-MMSE-based estimation error analysis is shown to be more accurate than other analyses (e.g., those that rely on Fisher information) when the data is noisy and/or when the data is small, i.e., when N is not much larger than P . This advantage is highly desirable in active learning settings and especially in affect detection for two reasons. First, in active learning settings, one often work with small problem sizes: in the initial stages of the active learning process, the classifier is highly inaccurate since it is only trained on a small number of observed labels; therefore, it can lead to an unreliable metric of informativeness which is the key to active learning methods. Second, in affect detection and a lot of other educational applications, the data is inherently noisy: state-of-the-art affect detectors can only achieve area under the receiver operating characteristic curve (AUC) values of around 0.7 after many empirical tweaks [5]. This accuracy is significantly lower than that in common classification tasks [13]. Moreover, inter-coder disagreement on a student's affective state can be high in some cases [27]; this disagreement is also reported in facial expression recognition-based affect detectors [3].

Therefore, we propose a new active learning method that uses the closed-form expression of the MSE of the L-MMSE estimator given in Eq. 1 to measure informativeness since it is reliable even for small and noisy data. Note that we do not use the L-MMSE estimator to estimate \mathbf{x} , but only its MSE to select the next observation. Specifically, we use the negative MSE as our metric of informativeness as

$$I(\mathbf{d}_i, \hat{\mathbf{x}}_{t-1}) = -\text{MSE}(\mathbf{D}_{\mathcal{O}_{t-1} \cup \{i\}}).$$

In other words, we select the t -th observation as the one corresponding to the feature vector \mathbf{d}_i that *minimizes* the resulting MSE, i.e.,

$$i_t = \underset{i \in \mathcal{U}_{t-1}}{\text{argmin}} \text{MSE}(\mathbf{D}_{\mathcal{O}_{t-1} \cup \{i\}}),$$

where $\mathbf{D}_{\mathcal{O}_{t-1} \cup \{i\}} = [\mathbf{D}_{\mathcal{O}_{t-1}}^T, \mathbf{d}_i^T]^T$.

Since the MSE is independent on the observations \mathbf{y} , the L-MMSE-based active learning method is likely more robust than all existing methods that rely on \mathbf{y} , especially during the initial stage of the active learning process when the number of observations is small. Therefore, it is likely to be highly

effective in real-world QFO and especially affect detection settings. This intuition is confirmed by our experiments in Section 4.

In practice, the MSE can be computed very efficiently since the inverse of the matrix \mathbf{C}_y^{-1} only needs to be computed once in every iteration; we do not need to invert it for every potential observation added to the current set of observations. For simplicity of exposition, we temporarily drop the subscripts and use \mathbf{D} and \mathbf{d} to denote the current feature matrix and the feature vector for a possible new observation. The new matrix \mathbf{C}'_z is given by

$$\mathbf{C}'_z = \begin{bmatrix} \mathbf{D} \\ \mathbf{d}^T \end{bmatrix} \mathbf{C}_x [\mathbf{D}^T \ \mathbf{d}] = \begin{bmatrix} \mathbf{C}_z & \mathbf{D} \mathbf{C}_x \mathbf{d} \\ \mathbf{d}^T \mathbf{C}_x \mathbf{D}^T & \mathbf{d}^T \mathbf{C}_x \mathbf{d} + 1 \end{bmatrix}.$$

Now, the new matrix \mathbf{C}'_y is given by

$$\begin{aligned} \mathbf{C}'_y &= \frac{2}{\pi} \sin^{-1} \left(\text{diag}(\text{diag}(\mathbf{C}'_z)^{-1/2}) \mathbf{C}'_z \text{diag}(\text{diag}(\mathbf{C}'_z)^{-1/2}) \right) \\ &= \frac{2}{\pi} \sin^{-1} \left(\begin{bmatrix} \text{diag}(\text{diag}(\mathbf{C}_z)^{-1/2}) & \mathbf{0} \\ \mathbf{0}^T & (\mathbf{d}^T \mathbf{C}_x \mathbf{d} + 1)^{-1/2} \end{bmatrix} \right. \\ &\quad \cdot \begin{bmatrix} \mathbf{C}_z & \mathbf{D} \mathbf{C}_x \mathbf{d} \\ \mathbf{d}^T \mathbf{C}_x \mathbf{D}^T & \mathbf{d}^T \mathbf{C}_x \mathbf{d} + 1 \end{bmatrix} \\ &\quad \cdot \left. \begin{bmatrix} \text{diag}(\text{diag}(\mathbf{C}_z)^{-1/2}) & \mathbf{0} \\ \mathbf{0}^T & (\mathbf{d}^T \mathbf{C}_x \mathbf{d} + 1)^{-1/2} \end{bmatrix} \right) \\ &= \begin{bmatrix} \mathbf{C}_y & \mathbf{c} \\ \mathbf{c}^T & 1 \end{bmatrix}, \end{aligned}$$

where $\mathbf{c} = \frac{2}{\pi} \sin^{-1} \left((\text{diag}(\mathbf{C}'_z)^{-1/2}) \mathbf{D} \mathbf{C}_x \mathbf{d} (\mathbf{d}^T \mathbf{C}_x \mathbf{d} + 1)^{-1/2} \right)$. Now, using the block matrix inversion rule [17], we have

$$\mathbf{C}'_y{}^{-1} = \begin{bmatrix} \mathbf{C}_y^{-1} + h \mathbf{g}^T \mathbf{C}_y \mathbf{g} & h \mathbf{g} \\ h \mathbf{g}^T & h \end{bmatrix},$$

where $\mathbf{g} = -\mathbf{C}_y^{-1} \mathbf{c}$ and $h = \frac{1}{1 - \mathbf{c}^T \mathbf{C}_y^{-1} \mathbf{c}}$. Now, the new matrix \mathbf{E}' is given by

$$\begin{aligned} \mathbf{E}' &= \left(\frac{2}{\pi}\right)^{1/2} \begin{bmatrix} \text{diag}(\text{diag}(\mathbf{C}_z)^{-1/2}) & \mathbf{0} \\ \mathbf{0}^T & (\mathbf{d}^T \mathbf{C}_x \mathbf{d} + 1)^{-1/2} \end{bmatrix} \\ &\quad \cdot \begin{bmatrix} \mathbf{D} \\ \mathbf{d}^T \end{bmatrix} \mathbf{C}_x = \begin{bmatrix} \mathbf{E} \\ \mathbf{e}^T \end{bmatrix}, \end{aligned}$$

where $\mathbf{e} = \left(\frac{2}{\pi}\right)^{1/2} (\mathbf{d}^T \mathbf{C}_x \mathbf{d} + 1)^{-1/2} \mathbf{D} \mathbf{C}_x \mathbf{d}$. Therefore, plugging all of the above into Eq. 1 and some algebra, we get an expression for the new MSE after adding a new observation with feature value vector \mathbf{d}_i as

$$\begin{aligned} \text{MSE}' &= \text{tr}(\mathbf{C}_x) - \text{tr}(\mathbf{E}'^T \mathbf{C}'_y{}^{-1} \mathbf{E}') = \text{tr}(\mathbf{C}_x) \\ &\quad - \text{tr} \left(\begin{bmatrix} \mathbf{E}^T & \mathbf{e} \end{bmatrix} \begin{bmatrix} \mathbf{C}_y + h \mathbf{g}^T \mathbf{C}_y^{-1} \mathbf{g} & h \mathbf{g} \\ h \mathbf{g}^T & h \end{bmatrix} \begin{bmatrix} \mathbf{E} \\ \mathbf{e}^T \end{bmatrix} \right) \\ &= \text{tr}(\mathbf{C}_x) - \text{tr}(\mathbf{E}^T \mathbf{C}_y^{-1} \mathbf{E}) - h \text{tr}(\mathbf{E}^T \mathbf{g}^T \mathbf{C}_y^{-1} \mathbf{g} \mathbf{E}) \\ &\quad - 2h \text{tr}(\mathbf{E}^T \mathbf{g} \mathbf{e}^T) - h \text{tr}(\mathbf{e} \mathbf{e}^T) \\ &= \text{MSE} - h(\|\mathbf{E}^T \mathbf{g} + \mathbf{e}\|_2^2), \end{aligned} \quad (2)$$

where the reduction in MSE induced by making a new observation is given by the term $h(\|\mathbf{E}^T \mathbf{g} + \mathbf{e}\|_2^2)$. Therefore, we can obtain the new MSE without having to explicitly calculate $\mathbf{C}'_y{}^{-1}$ for every possible new observation. In our experiments, we found that this implementation speeds up the L-MMSE-based active learning method by 10 to 100 times, resulting in an empirical computational complexity

that is lower than most existing active learning methods except uncertainty sampling.

4. EXPERIMENTAL RESULTS

We now perform a series of experiments on a real-world student affect dataset to explore the effectiveness of active learning methods. We start by adopting standard experimental protocols for active learning under several different settings and then present a simple example to help us understand the conditions under which active learning methods are the most effective.

4.1 Student Affect Dataset

We use an existing dataset for building sensor-free affect detectors collected in real classrooms² [5]. The dataset consists of 3,109 observations, each observation contains i) a student's affective state label during a 20-second observation interval in real classrooms and ii) a set of 88 features that summarizes their activities within ASSISTments during this time interval. These features include the time each student spent on practice items, the number of hints they seek, and the correctness of their responses. We keep observations where the student is labeled as being in one of the four affect states under BROMP: bored, confused, engaged concentration, and frustrated. We leave out the few observations where the human coder indicates that either the student is not in any of the four states or that they are not sure what state the student is in. Engaged concentration is the most frequent state among the four, which occurs about 82% of the time.

Since we focus on logistic regression-based affect detectors in this paper, we need to construct a binary classification problem by detecting the presence of one of the four affective states. We start by building a detector of the engaged concentration affective state since it is the most common among the four states.

4.2 Baseline Active Learning Methods

We test four different active learning methods in our experiments: i) our L-MMSE-based active learning method, (ii) uncertainty sampling (US) [24], as introduced in Section 3.1, (iii) expected variance reduction (EVR) [40], which selects the next observation as the one that results in the largest reduction of the variance of the classifier, and (iv) model change (MC) [7], which selects the observation that changes the classifier's parameters the most. We also use random sampling (Random), which randomly selects the next observation, as the baseline method to simulate the round-robin observation schedule followed in real classrooms when the dataset was collected. We do not test another popular active learning method, expected error reduction [31], since it has very high computational complexity and does not outperform other methods in several preliminary experiments.

4.3 Engaged Concentration Detection

We start by testing active learning methods for a detector of engaged concentration vs. other affective states.

4.3.1 Experimental setup

²This dataset is taken from <http://tiny.cc/affectdata>

We use cross validation to test the performance of active learning methods on the ASSISTments student affect dataset. We use two different settings for cross validation: we split the dataset at both the observation level (where each observation is regarded as a stand alone instance) and the student level (where all observation on a student is considered as an instance). We randomly select 20% and 10% of all instances as the test and validation sets, respectively, and use the rest as the training set. The test set is used to evaluate the predictive quality of the trained classifier, using the area under the receiver operating characteristic curve (AUC) metric [18]. This metric takes value in $[0, 1]$ and larger values indicate higher predictive quality.

We start by randomly selecting an initial batch of $M \in \{20, 100, 500\}$ observations (with both student activity feature vector and affective state label for each observation) from the training set; we then use them to train a base logistic regression classifier and use it as our initial affect detector. This experimental setting enables us to study the effectiveness of active learning methods when the amount of prior data available to the detector varies. Although the L-MMSE-based analysis is based on probit regression, we use the more widely-adapted logistic regression to test its robustness against model mismatch. The base classifier is trained using accelerated gradient descent [26] implemented in TensorFlow³ with a $P \times 1$ zero-vector as the initializer. We do not regularize the logistic regression classifier and instead use the validation set to decide when to terminate the training process and avoid overfitting. Specifically, after each (accelerated) gradient descent step, we evaluate the current detector on the validation set, and stop once its predictive quality stops improving (as measured by AUC).

Then, in each iteration of the active learning process, we select the next observation from the remaining ones in the training set according to their feature values, for each active learning method. We then add this new observation (both its feature vector and label) to the current batch and re-train the affect detector, using the previous estimate of the regression coefficients as the initializer. We then calculate the AUC of the re-trained affect detector on the test set. We repeat these steps for a total of 50 additional observations; using more data points is unnecessary since i) we found that using 50 additional observations is enough to summarize the behavior of each active learning method and ii) the performance of the affect detector will converge to the same end point for each active learning method, after going through the entire training set. We also repeat our experiment 100 times and use a different random split of the full dataset and a different initial batch of observations each time. We then report the average results over these repetitions.

4.3.2 Results and discussion

Figure 1 plots the AUC values of the trained affect detectors on the held-out test set vs. the number of additional observations, for all active learning methods on the student affect dataset, using observation-level cross validation. We see that most active learning methods, except EVR, generally outperforms random observation selection when the quality of the affect detector is limited by the amount of data it

³<https://www.tensorflow.org/>

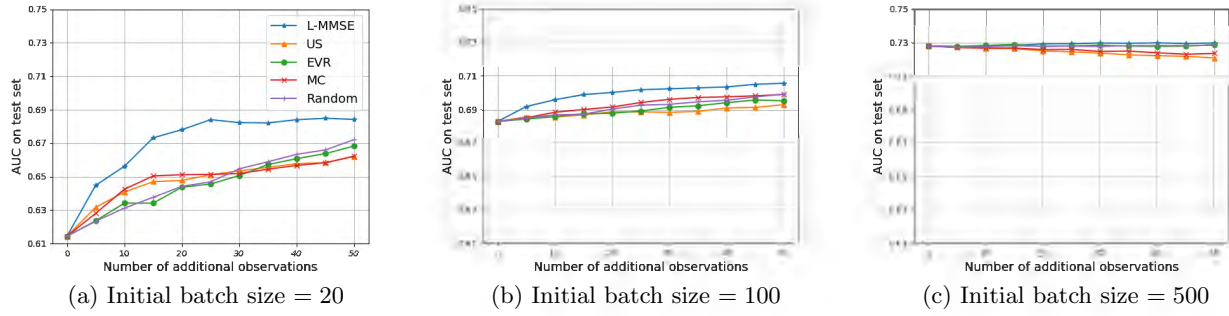


Figure 1: Comparison between different active learning methods for engaged concentration detection with observation-level cross validation. Most active learning methods, especially our L-MMSE-based active learning method, are effective at small initial batch sizes. This advantage over random observation selection diminishes as the quality of the detector saturates when a large number of observations is made.

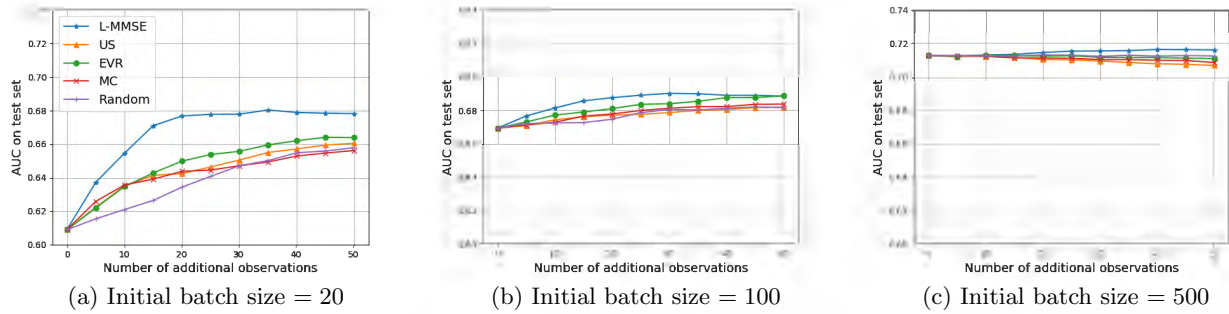


Figure 2: Comparison between different active learning methods for engaged concentration detection with student-level cross validation. The behavior of active learning methods remain largely the same as observation-level cross validation: they are most effective when the affect detector is trained on few observations.

sees (when it is trained on no more than 50 observations). Our L-MMSE-based method significantly outperforms every other method in this setting. As a concrete example, with 25 additional observations added to the 20 observations in the initial batch, the L-MMSE active learning method results in an AUC of 0.685 on the test set, while no other method result in a detector that has an AUC above 0.65. This result suggests that the L-MMSE-based active learning method excels at picking out observations that are crucial to the affect detector immediately, despite the detector’s limited predictive quality; its performance in this setting is impressive since the number of features is quite large ($P = 88$), which is even more than the number of observations in Figure 1(a). Moreover, to reach an AUC value of 0.685 on the test set, the L-MMSE-based active learning method only needs 45 total observations; no other active learning method can achieve this predictive quality even with 70 total observations. This result suggests that, by directing human experts at making observations that are more meaningful to the affect detector, active learning methods can potentially improve the quality of the data without requiring more human effort.

We demonstrate the statistical significance of our results using Student’s t-test. Table 1 shows the p-values for rejecting the null hypothesis that the best performing active learning method (L-MMSE) over random observation selection, with

No. of observations	20	30	40	70	100
p-value	3×10^{-3}	2×10^{-7}	2×10^{-9}	6×10^{-3}	4×10^{-1}

Table 1: Statistical significance of the advantage active learning (the L-MMSE-based method) exhibits over random observation selection. Active learning methods are significantly better at the initial stage of the affect observation process.

an initial batch size of $M = 20$. We see that initially, when the affect detector is not highly accurate, active learning has a significant advantage over random observation selection.

As the size of the initial batch increases ($M = 100$) and the quality of the initial affect detector improves, the advantage of the L-MMSE-based active learning method over random observation selection drops and eventually diminishes when $M = 500$. This result is not surprising since with 500 initial observations, the performance of the affect detector already saturates (the AUC on the test set after training on the entire training set is 0.74, which is consistent with the values reported in [5]). However, even in this case, the L-MMSE-based active learning method still provides some improvement compared to random observation selection (about 0.01 AUC on the test set with 100 to 150 observations in Figure 1(b)). We note that this advantage is not statistically significant

(see Table 1), which is not surprising since the quality of the affect detector improves very slowly after the first 50 observations, leaving very little room for active learning to show its effectiveness.

Perhaps surprisingly, no active learning method except our L-MMSE-based method consistently outperforms random observation selection, even when the initial batch size is small. When the initial batch size is large ($M = 500$), US and MC even leads to worse affect detectors, although we suspect that the performance degradation in that case is due to randomness in cross validation not being sufficiently smoothed out rather than a poor affect detector. These results confirm our intuition that active learning methods designed for general-purpose classification tasks are not well-suited to affect detection, especially when the data size is small during the initial stage of the data collection process.

Figure 2 plots the AUC values of the trained affect detectors on the held-out test set vs. the number of additional observations for all active learning methods, using student-level cross validation. The results largely remain the same compared to observation-level cross validation. Overall, there is a small drop of about 0.01 in test set AUC, confirming the intuition that it is harder for affect detectors to generalize to unseen students than to generalize to unseen observations from current students. However, the L-MMSE-based active learning method still (perhaps even more) consistently outperforms other active learning methods and random observation selection. As a concrete example, with only 10 additional observations in addition to an initial batch of 20 observations, the L-MMSE-based active learning method achieves an AUC of 0.655 on the test set; the other active learning methods and random observation selection achieve AUC values 0.635 and 0.62, respectively. In this case, the effectiveness of using active learning methods (especially our L-MMSE-based method) to identify informative observations and use them to improve affect detection is obvious.

Our experimental results also suggest that there is a lot of redundancy in the ASSISTments student affect dataset. As we discussed above, the quality of the affect detectors saturates after training on about 500 observations. Consider that the entire training set contains more than 2,100 observations, it seems that the majority of them do not significantly contribute to the quality of the resulting affect detector. This discovery further emphasizes the need of using smarter ways to collect higher-quality data; see Section 5 on a detailed discussion of how to use active learning methods to possibly improve data quality in practice.

4.4 Detection of Other Affective States

We now test the effectiveness of active learning methods for the detection of the other three affective states in BROMP: bored, confused, and frustrated.

4.4.1 Experimental setup

Since these affective states are rare (bored occurs about 10% of the time, while confused and frustrated each occur about 4% of the time) in the ASSISTments dataset, prior work [5, 28] uses resampling to balance among the affective states. Specifically, these works build training datasets that contain roughly equal numbers of observations corresponding

to each affective state by resampling from the original training set; after affect detectors are trained on the resampled training dataset, they are then evaluated on the original, non-resampled test set.

We do not use the resampling technique since our goal is to simulate the actual affect observation setting in real-world classrooms, where the four affective states are naturally unbalanced. Therefore, we use same experimental setting as before, except that we have to resort to larger initial batch sizes to ensure that at least a few rare affective states occur in the initial batch. In our experiments, we found that using an initial batch size of $M = 100$ is sufficient.

4.4.2 Results and discussion

Figure 3 plots the AUC values of the trained detectors on the held-out test set vs. the number of additional observations, for the affective states of bored, confused, and frustrated. We used different y-axis ranges in each of the three subplots to enhance contrast since for the confused and frustrated states, the improvement in the quality of the detectors as more observations are made is small. We see that active learning methods, especially our L-MMSE-based method, can still generally outperform random observation selection in most cases (especially for the bored state). However, this advantage is much smaller for these infrequent affective states compared to engaged concentration. For the detection of confusion, two of the active learning methods (US and MC) consistently underperform random observation selection, while our L-MMSE-based method shows some improvement only initially. The only active learning method that performs on-par with random observation selection is the EVR method. One possible explanation is that for harder-to-detect affective states like the confused state, the quality of the classifier is quite low (the final AUC on the test set is only 0.67), which leaves little room for active learning to show its effectiveness.

We now present a simple example to give us some insights on the conditions under which active learning methods are most effective in affect detection. Figure 4 compares the portion of observations selected by an active learning method (US) that actually correspond to an infrequent affect (we used the bored state as an example) to that of random observation selection. We see that after using the initial batch of observations to build a (low-quality) detector, active learning methods can quickly use it to select the observations that actually correspond to the infrequent target affect. Specifically, within the first 50 additional observations, US selects about 15 observations that correspond to the bored affective state, using only student activity features, as it deems these observations more informative; this portion (about 30%) is much higher than the overall portion of the bored state in the entire training set (about 10%). This behavior of US is consistent across all affective states except the confused state, where the portion of observations it selects that actually correspond to the confused state does not exceed the overall portion. In that case, active learning methods also fail to consistently outperform random observation selection, as shown in Figure 3(b). Therefore, active learning methods seem to be effective only if they can strike the right balance between observing different affective states that occur at different frequencies.

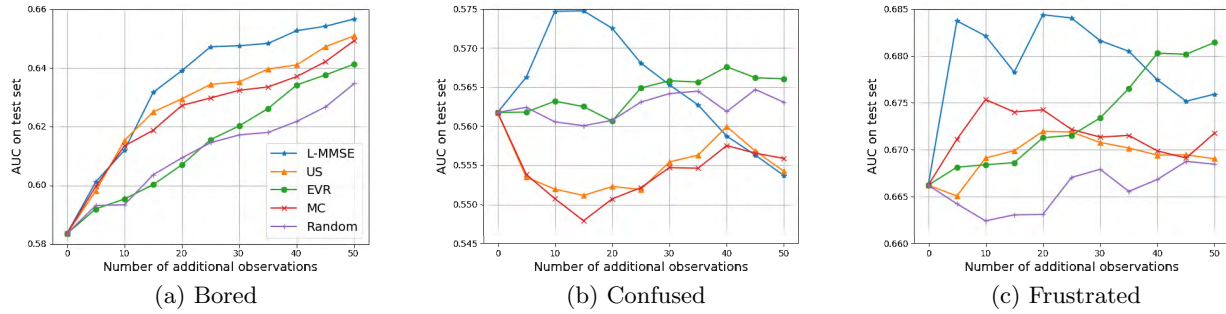


Figure 3: Comparison between different active learning methods for infrequent affective state detection (bored, confused, and frustrated). Active learning methods are generally effective for the bored and frustrated states but not the confused state. Their advantage over random observation selection for these states is smaller than that for engaged concentration detection.

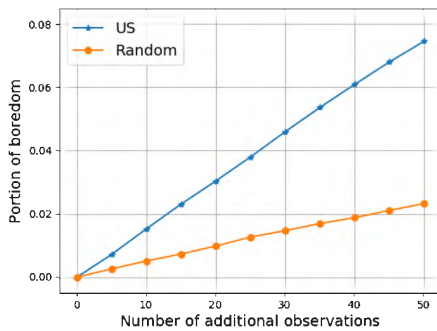


Figure 4: Portion of total infrequent affective state observations selected by an active learning method (US) versus random observation selection for boredom detection. Active learning methods can effectively select observations that actually correspond to the infrequent affect.

5. DEPLOYMENT IN CLASSROOMS

We now outline how to deploy active learning methods in real classrooms to improve the data collection efficiency for affect detection. Since active learning requires training affect detectors on-the-fly as new observations are made, there is a need to create a system that consists of three components. The first component is an interface to human observers making observations in classrooms; this interface i) suggests the human observer to observe a student at each observation interval, ii) collects their affect label on the student, and iii) send the label to the affect detector. The second component is a training paradigm for affect detectors that keeps updating the detector by re-training it after it receives each observed affect label and its corresponding feature vector. The third component is the active learning method that links the other two components together: it i) uses APIs to collect student activity data from ITSs and turn them into feature vectors, ii) selects the next observation that is the most informative to the current affect detector and sends its suggestion via the human observer interface.

There are several realistic considerations in such a system in order for it to be deployed in real classrooms. First, our

experiments (see Section 4.4) have shown that active learning methods are not as effective for affective states that occur infrequently (especially the confused state). Therefore, there is a need to explore more advanced active learning methods that take class imbalance into account [11]. Second, experienced human observers may have their own understanding of the informativeness of an observation; such understanding can also be highly valuable to machine learning-based affect detectors. Therefore, the human observer interface should present an option that allows them to ignore the suggestion by active learning methods and instead propose which students to observe on their own. Third, fairness among different student subgroups [39] is critical; we want to ensure that each subgroup is well-observed in the data collection process. Therefore, there needs to be an exploration mechanism that checks whether a student subgroup is under-observed and limit active learning methods to only select among those students when that happens.

6. LIMITATIONS AND FUTURE WORK

In this paper, we have explored the problem of whether active learning methods can be used to increase the efficiency of the affective state label collection process for the development of sensor-free affect detectors. Using an existing student affect dataset collected from ASSISTments, we have shown that active learning methods are indeed effective at making observations that are the most informative to the affect detector; therefore, it can reduce the number of observation needed for the detector to reach a certain quality under most settings. We also proposed a new active learning method that is especially effective for small and noisy data; experimental results show that it outperforms existing active learning methods. At the end, we outlined how to deploy these methods in real-world systems to improve the quality of the data to be collected and discussed several necessary considerations under practical constraints.

Despite the effectiveness of active learning methods, especially our L-MMSE-based method, our work has several limitations and can be extended in many different ways. First, our experimental setting for active learning does not perfectly reflect the actual affective state observation process in real classrooms. In our experimental setting, we select the next observation from all available observations left in the

training set, which was collected in many classroom sessions over a long period of time. In practice, when a human observer is making observations in real classrooms, we can only select an observation among the students in class; the most informative observation among these students is generally less informative than the most informative observation possible. Therefore, the benefit active learning methods bring to real-world affect label collection may not be as much as what we have shown in our experiments.

Second, the affect detectors we have studied in this paper are only for detecting the presence of a particular affective state, e.g., bored vs. not bored; it cannot jointly detect all possible affective states. The reason we did so is to test as many active learning methods as possible since most of them are designed only for binary classification. Unfortunately, the most effective active learning method for affect detection in our experiments (our L-MMSE-based method) only applies to binary classification tasks. Therefore, for real-world affect detection problems that are multi-class classification problems, we will extend our method so that it can be applied to multinomial logistic regression instead of binomial logistic regression.

Third, state-of-the-art affect detectors use neural networks rather than logistic regression as their base classifier [5]. While some active learning methods (e.g., uncertainty sampling) can be easily extended to neural networks, others (e.g., our L-MMSE-based method, variance reduction methods, and methods based on model change) cannot since they are either theoretically grounded in binary regression or become computationally intractable. Fortunately, the L-MMSE estimation framework encapsulates all the common nonlinearities used in today's state-of-the-art neural network architectures, including the hyperbolic tangent and rectified linear nonlinearities [13]. Therefore, we will extend the L-MMSE-based active learning method proposed in this paper to leverage neural networks as the base classifier.

Fourth, the workflow we outlined for the deployment of active learning in a real-world system in Section 5 presents a time mismatch challenge. In order to select an observation that the human observer should observe, we need access to the corresponding student activity feature vector; these feature values, however, are not available until the end of the observation time interval since many features summarize a student's activity during the entire period. When the teacher receives a suggestion to observe a certain student, this suggestion will be based on the student's activities during the last observation interval, which may not be the most informative observation during the current observation interval. Therefore, we will need to perform a thorough analysis of the coherence in student activity and affect over time to validate the feasibility of deploying active learning in real-world systems for affect label collection.

Finally, the essence of using active learning for affect detection is to leverage the judgement a machine learning-based detector makes on how sure it is about the affective state of a student. Simultaneously, human observers who are trained to make observations in classrooms have their own judgements on how sure they are about a student's affective state. Therefore, comparing the two sets of judgements may lead

to deeper insights on how humans perceive affect. Moreover, there is an intrinsic mismatch between the two sets of judgements since one is based on a set of activity features in ITSs while the other is based on observations of activity, gesture, and facial expressions. Therefore, comparing the two sets of judgements may also lead to an analysis of the extent to which the activity features can capture student affect; these insights can potentially help us to design better student activity features or even lead to better ITS designs.

7. REFERENCES

- [1] V. Aleven, F. Xhakaj, K. Holstein, and B. M. McLaren. Developing a teacher dashboard for use with intelligent tutoring systems. In *Proc. International Workshop on Teaching Analytics at the European Conference on Technology Enhanced Learning*, pages 15–23, Sep. 2016.
- [2] I. Arroyo, B. P. Woolf, W. Bureson, K. Muldner, D. Rai, and M. Tai. A multimedia adaptive tutoring system for mathematics that addresses cognition, metacognition and affect. *International Journal of Artificial Intelligence in Education*, 24(4):387–426, Dec. 2014.
- [3] A. M. Aung and J. Whitehill. Harnessing label uncertainty to improve modeling: An application to student engagement recognition. In *IEEE International Conference on Automatic Face & Gesture Recognition*, pages 166–170, May 2018.
- [4] R. S. Baker, J. Ocumpaugh, S. M. Gowda, A. M. Kamarainen, and S. J. Metcalf. Extending log-based affect detection to a multi-user virtual environment for science. In *Proc. International Conference on User Modeling, Adaptation, and Personalization*, pages 290–300, July 2014.
- [5] A. F. Botelho, R. S. Baker, and N. T. Heffernan. Improving sensor-free affect detection using deep learning. In *Proc. International Conference on Artificial Intelligence in Education*, pages 40–51, July 2017.
- [6] W. Bureson. Affective learning companions: Strategies for empathetic agents with real-time multimodal affective sensing to foster meta-cognitive and meta-affective approaches to learning, motivation, and perseverance. Technical report, Ph.D. Thesis, Massachusetts Institute of Technology, 2006.
- [7] W. Cai, Y. Zhang, Y. Zhang, S. Zhou, W. Wang, Z. Chen, and C. Ding. Active learning for classification with maximum model change. *ACM Transactions on Information Systems*, 36(2):15, Sep. 2017.
- [8] S. Craig, A. Graesser, J. Sullins, and B. Gholson. Affect and learning: An exploratory look into the role of affect in learning with AutoTutor. *Journal of Educational Media*, 29(3):241–250, Oct. 2004.
- [9] M. Csikszentmihalyi. *Flow: The Psychology of Optimal Performance*. HarperCollins Publishers, 1990.
- [10] S. D'Mello, B. Lehman, J. Sullins, R. Daigle, R. Combs, K. Vogt, L. Perkins, and A. Graesser. A time for emoting: When affect-sensitivity is and isn't effective at promoting deep learning. In *Proc. International Conference on Intelligent Tutoring Systems*, pages 245–254, June 2010.
- [11] S. Ertekin, J. Huang, L. Bottou, and L. Giles. Learning on the border: Active learning in imbalanced data classification. In *Proc. ACM conference on Conference*

- on *Information and Knowledge Management*, pages 127–136, Nov. 2007.
- [12] K. Forbes-Riley and D. J. Litman. Adapting to student uncertainty improves tutoring dialogues. In *Proc. International Conference on Artificial Intelligence in Education*, pages 33–40, 2009.
 - [13] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
 - [14] N. T. Heffernan and C. L. Heffernan. The ASSISTments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, Dec. 2014.
 - [15] S. C. Hoi, R. Jin, J. Zhu, and M. R. Lyu. Batch mode active learning and its application to medical image classification. In *Proc. International Conference on Machine Learning*, pages 417–424, June 2006.
 - [16] F. Hollands and I. Bakir. Efficiency of automated detectors of learner engagement and affect compared with traditional observation methods. Technical report, New York, NY: Center for Benefit-Cost Studies of Education, Teachers College, Columbia University, Aug. 2015.
 - [17] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.
 - [18] J. Huang and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3):299–310, Mar. 2005.
 - [19] S. Karumbaiah, R. Lizarralde, D. Alessio, B. P. Woolf, I. Arroyo, and N. Wixon. Addressing student behavior and affect with empathy and growth mindset. In *Proc. International Conference on Educational Data Mining*, pages 96–103, July 2017.
 - [20] B. Kort, R. Reilly, and R. W. Picard. An affective model of interplay between emotions and learning: Reengineering educational pedagogy-building a learning companion. In *Proc. IEEE International Conference on Advanced Learning Technologies*, pages 43–46, Aug. 2001.
 - [21] A. S. Lan, M. Chiang, and C. Studer. An estimation and analysis framework for the Rasch model. In *Proc. International Conference on Machine Learning*, pages 2889–2897, July 2018.
 - [22] A. S. Lan, M. Chiang, and C. Studer. Linearized binary regression. In *Proc. Conference on Information Sciences and Systems*, pages 1–6, Mar. 2018.
 - [23] B. Lehman, M. Matthews, S. D’Mello, and N. Person. What are you feeling? Investigating student affective states during expert human tutoring sessions. In *Proc. International Conference on Intelligent Tutoring Systems*, pages 50–59, June 2008.
 - [24] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proc. ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, July 1994.
 - [25] M. Miserandino. Children who do well in school: Individual differences in perceived competence and autonomy in above-average children. *Journal of Educational Psychology*, 88(2):203–214, June 1996.
 - [26] Y. Nesterov. Gradient methods for minimizing composite objective function. Technical report, Université Catholique de Louvain, Sep. 2007.
 - [27] J. Ocuppaugh, R. S. Baker, and M. M. T. Rodrigo. Baker Rodrigo Ocuppaugh monitoring protocol (BROMP) 2.0 technical and training manual. Technical report, New York, NY and Manila, Philippines: Teachers College, Columbia University and Ateneo Laboratory for the Learning Sciences, May 2015.
 - [28] Z. A. Pardos, R. S. Baker, M. San Pedro, S. M. Gowda, and S. M. Gowda. Affective states and state tests: Investigating how affect and engagement during the school year predict end-of-year learning outcomes. *Journal of Learning Analytics*, 1(1):107–128, May 2014.
 - [29] M. O. Pedro, R. Baker, A. Bowers, and N. Heffernan. Predicting college enrollment from student interaction with an intelligent tutoring system in middle school. In *Proc. International Conference on Educational Data Mining*, pages 177–184, July 2013.
 - [30] J. Roschelle, M. Feng, R. F. Murphy, and C. A. Mason. Online mathematics homework increases student achievement. *AERA Open*, 2(4):1–12, Oct. 2016.
 - [31] N. Roy and A. McCallum. Toward optimal active learning through Monte Carlo estimation of error reduction. In *Proc. International Conference on Machine Learning*, pages 441–448, June 2001.
 - [32] J. L. Sabourin and J. C. Lester. Affect and engagement in game-based learning environments. *IEEE Transactions on Affective Computing*, 5(1):45–56, Jan. 2014.
 - [33] O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. In *Proc. International Conference on Learning Representations*, pages 1–13, May 2018.
 - [34] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, Nov. 2012.
 - [35] B. Taylor, A. Dey, D. Siewiorek, and A. Smailagic. Using physiological sensors to detect levels of user frustration induced by system delays. In *Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 517–528, Sep. 2015.
 - [36] Y. Wang, N. T. Heffernan, and C. Heffernan. Towards better affect detectors: Effect of missing skills, class features and common wrong answers. In *Proc. International Conference on Learning Analytics and Knowledge*, pages 31–35, Mar. 2015.
 - [37] M. Wixon, I. Arroyo, K. Muldner, W. Burleson, D. Rai, and B. Woolf. The opportunities and limitations of scaling up sensor-free affect detection. In *Proc. International Conference on Educational Data Mining*, pages 145–152, July 2014.
 - [38] Y. Yang and M. Loog. A benchmark and comparison of active learning for logistic regression. *arXiv preprint arXiv:1611.08618*, 2016.
 - [39] S. Yao and B. Huang. Beyond parity: Fairness objectives for collaborative filtering. In *Proc. Conference on Advances in Neural Information Processing Systems*, pages 2921–2930, Dec. 2017.
 - [40] K. Yu, J. Bi, and V. Tresp. Active learning via transductive experimental design. In *Proc. International Conference on Machine Learning*, pages 1081–1088, June 2006.

Toward Data-Driven Example Feedback for Novice Programming

Rui Zhi
North Carolina State Univ.
Raleigh, NC, USA
rzhi@ncsu.edu

Nicholas Lytle
North Carolina State Univ.
Raleigh, NC, USA
nalytle@ncsu.edu

Samiha Marwan
North Carolina State Univ.
Raleigh, NC, USA
samarwan@ncsu.edu

Thomas W. Price
North Carolina State Univ.
Raleigh, NC, USA
twprice@ncsu.edu

Yihuan Dong
North Carolina State Univ.
Raleigh, NC, USA
ydong2@ncsu.edu

Tiffany Barnes
North Carolina State Univ.
Raleigh, NC, USA
tmbarnes@ncsu.edu

ABSTRACT

Viewing worked examples before problem solving has been shown to improve learning efficiency in novice programming. Example-based feedback seeks to present smaller, adaptive worked example steps during problem solving. We present a method for automatically generating and selecting adaptive, example-based programming feedback using historical student data. Our data-driven feature-based (DDF) example generation method automatically learns program features from data and selects example pairs based on when students complete each feature. We performed an experiment to compare three example generation methods: Student trace data, Data-Driven Features (DDF), and Expert examples. Two experts rated the quality of feedback for each generator, and they rated both the Expert and DDF example feedback as significantly more relevant to students' goals than the Student example feedback. However, there were no significant differences between the DDF and Expert examples. We compared these approaches to one that combined DDF with an Interactive Selection step (DDF-IS), where the user (in this case, an expert) selects their preferred data-driven feature before an example is selected. DDF-IS produced significantly more relevant examples than all other approaches, with significantly higher overall example quality than DDF. This suggests that our DDF approach allows more relevant examples to be selected than existing approaches, and that we may be able to leverage interactivity with the student to further improve example quality.

1. INTRODUCTION & BACKGROUND

Prior studies show that worked examples are an effective instructional support to help novices learn complex tasks [27, 30, 28, 8]. Sweller argues that "...for novices, learning via worked examples should be superior to learning via problem solving." [27]. In the domain of programming, researchers

also suggest using worked examples to teach novices [3, 31]. Empirical studies have shown that interleaving worked examples with similar practice problems is more effective than solving only equivalent programming problems by writing code, as students spent less time on training tasks and performed better on a posttest [30]. However, worked examples are traditionally only offered to students in between problem solving attempts [30, 16, 7, 6], and they do little to assist students when they have difficulty *during* problem solving. Based on the idea of worked examples, researchers have explored example-based feedback [9, 4, 13], which shows a correct piece of code to help students learn during problem solving, as a form of adaptive, on-demand support [9]. Similar to a high-level on-demand hint, example-based feedback demonstrates one step in a correct solution to the problem the student is working on, selected adaptively to match the student's code. Keuning et al. argue for the need for such feedback in their review on automated feedback generation for programming, saying "the very low percentage of tools that give code examples based on the student's actions is unfortunate, because studying examples has proven to be an effective way of learning" [15]. As Gross et al. argue, showing a relevant partial solution could impose less cognitive load and be easier to visually present to novices than showing a full solution [9]. Ichinco et al. found that novices have trouble transferring what they learned from similar problems to their own code [14]. Example-based feedback addresses this by providing example steps from the *same* problem the student is working on. Figure 1 presents a prototype of what example-based feedback might look like in a block-based novice programming environment. The example is presented with a "before" state, similar to the student's current code, and an "after" state that completes a desired feature, helping the student easily identify the purpose and outcome of a single solution step.

Existing example-based feedback systems may rely on a library of expert-authored examples [11, 14, 4], which can be costly to maintain, as instructors are unlikely to create new examples [12]. Additionally, the example code may show an example solution to a related problem [11] that requires students to transfer knowledge to solve their current problem, which may be challenging for weaker students who need more help [13]. To address these limitations, we present a

Rui Zhi, Samiha Marwan, Yihuan Dong, Nicholas Lytle, Thomas Price and Tiffany Barnes "Toward Data-Driven Example Feedback for Novice Programming" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 218 - 227

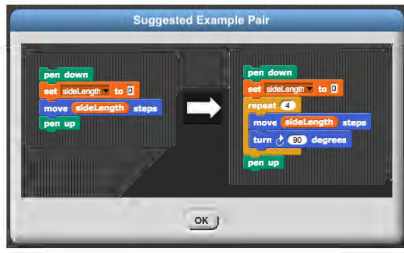


Figure 1: Example-based feedback prototype in iSnap.

data-driven method to create example-based feedback using historical student data. We focus on the domain of programming, where many problems have a vast space of possible solutions [24, 19], so a static set of examples is unlikely to be relevant to all students. To our knowledge, only Gross et al. have previously employed a data-driven method to derive examples adaptively to help students [9]. They found that examples derived from expert solutions are perceived by students as more helpful and help students make more solution improvements than those derived from student solutions. However, they simply presented the complete student solution which was most similar to the current student’s code (according to a distance metric), rather than trying to systematically identify and address students’ current programming goals. This suggests that existing evaluations of data-driven methods for example-based feedback have not explored the full potential of the approach. We hypothesize that with the innovations in this work, data-driven examples can be more adaptive than expert-authored ones, while still presenting correct and interpretable solution steps.

In this work, we present an approach to automatically detect students’ progress towards a solution, and suggest example-based feedback from historical student data. Our data-driven method first processes prior correct student solutions to discover meaningful features, labels the parts of code that contribute to each feature, and removes code that does not contribute to the solution, to create simple data-driven code examples that contain only the code needed for each feature. We then automatically label the student’s current code with a “feature state” representing the presence or absence of each data-driven feature needed for a correct solution. We then adaptively select example feedback that contains the same features as the current student code, and adds a new feature that is relevant to solving the current problem. In contrast, other example feedback systems show code from a related similar problem, requiring students to study the example and transfer what they learn to the current context [11].

We evaluated two data-driven methods to generate and select example-based feedback for historical student hint requests: Data-Driven Features (DDF), and Data-Driven Features with Interactive Selection (DDF-IS). We compared these methods against two baselines: (1) Expert-authored examples (Expert) and (2) examples generated naively from correct student solution traces, showing the code added between consecutive test runs (Student). The data-driven features (DDF) algorithm cleans prior students solutions and generates example pairs where the “start code” has the same features as the student’s code, and the “end code” adds a new feature that is not yet present in the student’s code.

The DDF with Interactive Selection (DDF-IS) approach explored the potential to improve algorithmic DDF example feedback selection by having a user interactively select the data-driven feature with which they want help, before the algorithm selects an example. To simulate this experience, we used an expert to select this feature, representing a best-case scenario for interactive selection.

We adapted a multidimensional data-driven hint evaluation rubric from previous work to evaluate the example-based feedback quality based on Relevance, Progress, Interpretability, and Similarity. Our findings showed both the Expert and DDF feedback were significantly more relevant to students’ goals than the Student feedback, but there were no significant differences between the DDF and Expert examples. This suggests that our DDF feedback can reasonably replace Expert-authored examples in situations where they are unavailable or difficult to scale. We also found that DDF-IS produced significantly more relevant examples than all other approaches, with the highest overall example quality, significantly higher than DDF. This suggests that in the best case, data-driven feedback may leverage interactivity with the student to further improve example quality.

The contributions of this paper are: 1) a data-driven algorithm capable of generating adaptive example feedback for students during programming, and 2) an initial evaluation showing that these adaptive examples can be more relevant than static, expert-authored examples.

2. METHOD

This work presents and evaluates a data-driven feature-based (DDF) method for generating and selecting example-based feedback (explained in Section 2.2). To evaluate our DDF approach, we generated example-based feedback for historical student help requests, and asked experts to evaluate the quality of each type of feedback, comparing against two baselines (explained in Section 2.3.1).

2.1 Dataset

Our dataset comes from iSnap [20], which extends the Snap! block-based programming environment with logging and on-demand, data-driven hint support. It logs student interactions with the system, including complete code snapshots after each edit. The data were collected during the Fall 2016 (F16), Spring 2017 (S17), and Fall 2017 (F17) semesters in an introductory computing course for non-majors, held at a research university¹. In each semester, the students completed 3 in-lab assignments with access to help from teaching assistants and 3 homework assignments independently. In this paper, we selected one homework assignment *Squirrel* for the example code generation and evaluation. In *Squirrel* (shown in Figure 2), students program a “sprite” to draw a spiraling square-like shape using loops, variables, arithmetic operators and a custom block (function). Common solutions for *Squirrel* contain 7-10 lines of code. The original dataset contains 57 (F16), 43 (S17), and 47 (F17) *Squirrel* assignment submissions. Since iSnap offers students on-demand, data-driven hints which may alter students’ problem-solving patterns, we exclude students who requests hints in the dataset

¹All datasets are available at <https://pslcdatashop.web.cmu.edu/Project?id=321>

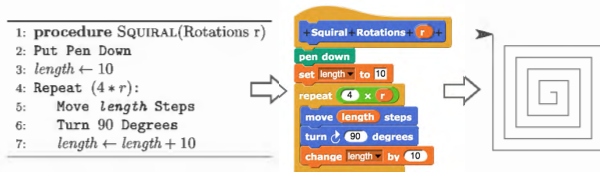


Figure 2: Squirrel pseudocode, Snap! code and output [34].

used for the example-based feedback generation. Our remaining data contained 38, 29, 39 code traces for the F16, S17, and F17 semesters, respectively. Each code trace contains the set of timestamped snapshots that comprise all of a student’s work on a problem.

2.2 Example-based Feedback Generation

We propose an algorithm to automatically generate data-driven, example-based feedback with the following high-level steps: 1) Extract a set of data-driven features from prior student code traces, which each describe a property of a correct solution. 2) Use the features to clean correct student code traces by removing extraneous code that does not contribute to a correct solution. 3) Generate pairs of example code snapshots that demonstrate how to complete a single feature. 4) Choose an appropriate, personalized example pair upon student request.

Our goal in this work is to perform a preliminary evaluation of our algorithm before implementing a user interface and evaluating its impact in practice. However, to contextualize our work, Figure 1 presents one way that such feedback could be presented to a student in iSnap when they request help. Our DDF example pairs consist of “start code,” similar to the student’s, and “end code” that demonstrates how to complete a single feature. Our design draws inspiration from a variety of theoretical and empirical sources, including cognitive load theory [29], Vygotsky’s Zone of Proximal Development [32], worked examples [27], learning from subgoals [16], and compare/contrast tasks [17].

Research on worked examples suggests that seeing examples which break a problem down into sequential steps (e.g. features) can be a more efficient way of learning than problem solving [27, 30, 28]. According to cognitive load theory, worked examples are effective because they lower the extraneous cognitive load (mental effort) imposed by the instructional materials. We designed our examples to present steps as a pair of “start” and “end” code, as previous studies have shown that comparing and contrasting examples is an effective learning activity in many domains [17, 26]. The example is adaptively selected to keep students in the Zone of Proximal Development [32] by starting with code similar to the student’s, which they can already understand, and scaffolding the completion of a new feature, which they cannot yet accomplish on their own. The work of Morrison et al. [16] suggests that programming examples that are broken into subgoals can help improve learning for novices. When examples are isomorphic to the problem solving task, as in our case, they found that it is most effective for students to label subgoals themselves, a feature we could easily incorporate into our example-based feedback.

2.2.1 Step 1: Data-driven Feature Generation

The goal of an example pair is to present how a meaningful self-contained portion of solution code, or *feature*, can be completed. An assignment may have students program multiple features, and the final correct solution should have all the correct features present. For example, in Squirrel (shown in Figure 2), a feature could be to move the sprite in a square shape, draw some figure on the screen, or repeat the spiral the correct number of times. In our previous work [34], we manually defined expert-authored features (shown in Table 1) in a systematic way, and we also implemented a data-driven algorithm to automatically identify code features from student solutions. Our results showed that many of the data-driven features were easily interpretable and closely matched the expert-authored features. The two methods also had moderate agreement on whether a given student was in the same state or different states.

The full procedure for data-driven feature extraction is given in [34], but we outline its high-level steps as follows:

- 1) **Preprocess student solutions:** Some student solutions may contain extraneous code or procedures that were used for testing or resetting the environment, which we attempt to remove before extracting features. Specifically, we used the SourceCheck algorithm [21] to identify and remove whole scripts and procedures that do not match any element of an expert-authored solution in the correct student solutions.
- 2) **Generate code shapes:** To identify common code patterns in correct student solutions, we extract a set of *code shapes*, or syntactic structures, from the solution code by converting students’ solution code into abstract syntax trees (ASTs) and then identify all pq-Gram subtrees [1] in each AST, to form our initial set of “code shapes.” This includes all code shapes from all correct solutions.
- 3) **Remove duplicates:** The initial set of code shapes may include very similar shapes, including some AST patterns that are subsets of others. Therefore, we remove these duplicates by measuring the co-occurrence of code shapes in all student code traces and keeping only the more specific code shape and discard the other duplicate if two code shapes almost always appear together in the same code.
- 4) **Identify decision shapes:** Due to varied problem solving strategies, some code shapes may not appear in every correct solution. For example, a Squirrel solution can either use nested repeat or a single repeat block to rotate the correct number of times (as shown in Figure 2), but not both. Therefore, we define a *decision shape* as a disjunction of code shapes, where almost all solutions contain exactly one of the component code shapes. A decision shape is present in a solution if any one of its code shapes exist in its code.
- 5) **Filter out uncommon code and decision shapes:** Since we are interested in using code and decision shapes to represent features of a correct solution, we keep only those shapes which appear in the vast majority of correct solutions, and filter out the rest.
- 6) **Form features:** Our goal is to define a small set of features that collectively represent a complete solution. However, the previous steps will generate tens to hundreds of code and decision shapes for a relatively simple problem like Squirrel. We therefore combine these smaller shapes into larger features using a form of hierarchical clustering. We iteratively combine any two features that most frequently

Table 1: Expert Features and Corresponding Data-Driven Features, as derived in [34].

Feature Name	Brief Description	Data-driven Analogue
E1. Procedure	Primary code inside of a procedure.	D1: Create a procedure OR a variable.
E2. Draw Anything	Able to draw anything on screen.	D8: Use a ‘repeat’ AND create a variable OR a parameter.
E3. Move ‘Square -like’	Able to move sprite in a square-like fashion.	D11: Have a ‘move’ AND a ‘turn’ in a ‘repeat’ AND have a ‘pen down.’
E4. Correctly Use Parameter	Correctly uses parameter within custom block.	D4: Have a ‘repeat’ inside a procedure.
E5. Repeat Correct # of Times	Repeats square-like movement correct number of times.	D5: Have a ‘multiply’ block with a variable OR two nested ‘repeats’.
E6. Move ‘Variably’	Movement is based on a variable not literal amount.	D10: Have a ‘move’ with a variable argument inside of a ‘repeat’.
E7. Move ‘Squirally’	Increase length to move for each side.	D7: Change a variable inside a ‘repeat’.

co-occur across student data until the size of the observed state-space defined by the features starts to decrease rapidly.

7) **Represent student code by feature vectors:** Once the features are formed, we can represent a student’s current code as a vector indicating the presence or absence of each feature. A student starts with a feature state of all 0s, and a correct solution should have all features present, resulting in a vector of all 1s.

2.2.2 Step 2: Cleaning Student Code

To extract good example pairs from prior student traces, we need to first remove excess blocks which do not contribute to a correct solution, as these may distract students and make examples harder to interpret. Identifying the excess blocks can be difficult, especially for intermediate partial solutions, since students can construct solutions in a large variety of ways [21, 24]. We address this by leveraging the features we defined in step 1 to exclude *irrelevant code*, which does not belong to any feature. Specifically, our cleaning procedure removes one node from the abstract syntax tree at a time (including all its children), then checks whether removing this node causes a currently completed feature to become incomplete. If removing a node “breaks” a feature, we assume that it is necessary and add it back; otherwise, we remove it. We also check to make sure removing the node does not break any code dependencies, such as deleting a variable declaration when the variable is used elsewhere. We iterate over every node in a recursive, breadth-first manner, starting from the root node. Once this iteration stops, it produces a cleaned partial solution, where all irrelevant code has been removed. We apply this cleaning procedure to *all* snapshots in correct solution traces. With well-defined features, this process can effectively clean a large variety of both partial and complete solutions, ensuring that all remaining code is useful. We use this process both for cleaning code and for extracting example pairs, as described in the next step.

2.2.3 Step 3: Extract Example Pairs

Our goal is to create a database of correct, meaningful, and self-contained example pairs to offer as feedback to students. Naively, we could extract a single example pair for each feature from each correct code trace, since each student completed each feature at least once. However, we want to generate as many example pairs as possible, so that the algorithm can adaptively select one that is similar to another

student’s code. We therefore developed a method to generate many “synthetic” example pairs, each consisting of a pair of code states $(c_0, c_1)_i$, from any cleaned student code trace. Recall that each pair should cleanly demonstrate the completion of exactly one feature by contrasting a “start code” state (c_0) and an “end code” state (c_1). The algorithm first extract one example pair each time a student completes a feature f_i , with c_1 defined as the snapshot right after f_i was completed, and c_0 as the snapshot right after the *prior* feature f_{i-1} was completed. We generate additional example pairs from each cleaned snapshot in a student solution trace with the following procedure. For each snapshot, the algorithm labels it as an end state, c_1 . It then removes exactly one data-driven feature from c_1 to create a c_0 , and together these form the example pair. This feature removal is accomplished using the code cleaning procedure described above; however, instead of removing irrelevant nodes, we use it to remove whole features. The algorithm first tries to remove one leaf node, l_i , at a time. Since the snapshot has already been cleaned, removing this node will either create an invalid code state, or cause a feature to become incomplete. In the later case, the cleaning procedure was run to remove all other code associated with the removed feature. The resulting cleaned code becomes the c_0 for the example pair, and our cleaning procedure guarantees that c_0 will have exactly one less feature than c_1 . The $(c_0, c_1)_i$ pair is added to a list which stores our example pairs. The algorithm then repeats this process recursively on c_0 , which becomes the c_1 for new example pairs, until no new pairs can be generated. In this case, we generate many example pairs per *snapshot* in a solution trace. While some are redundant, many are unique.

2.2.4 Step 4: Select an Example Code Pair

When a student requests help, we aim to provide them with the most appropriate example pair in our database as feedback. We define two ways that we can identify this example pair: 1) a **Data-Driven Features (DDF)** approach, using an algorithm to select the best example pair, or 2) an **Data-Driven Features with Interactive Selection (DDF-IS)** approach, giving the student the information needed to select an example pair. We first consider the DDF approach, in which we attempt to select the example pair which is most similar to the current student’s code. Based on this selection criteria, the selected example code pair should be very similar to the student’s code, with the goal of minimizing

the effort needed to process the “start code” and allowing the student to focus on the feature demonstrated by the example pair. In this study, for the DDF feedback, we selected an appropriate example code pair as follows:

Generate proper example pair candidates: To filter out inappropriate pair candidates based on the student’s completed features, we select only those example pairs whose “start code” has the same features as the current student code, from the example pair lists (generated in Section 2.2.3). If we cannot find any, we sort the example pairs by the Hamming distance between the feature states of “start code” and student’s current code. Then we select the example pairs with the closest start-state to student’s current code. If there are multiple example pairs available (they all have the same start state), we use the SourceCheck algorithm to sort the example pairs based on the similarity between the “start code” and student’s current code.

Select one example pair from candidates: We iterate over the example pair candidates and select exactly one pair, which accomplishes a feature that the student has not finished yet, preferring features that the majority of students in a state similar to the current student will take next.

To generate *interactive* example-based feedback, we need a way to communicate to the student which features have an available example that they can request. By default, our features are unlabeled (having been generated automatically from data), but with a small amount of instructor effort (about 3 minutes), they can be labeled. By viewing the code shapes required by each feature, an instructor familiar with the problem can generate a short, human-readable description, such as “move the sprite using a variable”, or “make the sprite move further each time”. These options can then be shown to a student. Once the student has selected a feature for an example, we select an appropriate example pair that accomplishes that feature with start code matching the student’s current feature state. If there are multiple options, we use the same criteria as in DDF: select the example pair with the most similar start state to the student’s code, which contains a proper subset of the student’s features. Note that unlike in DDF, a student *could* request an example for a feature that they have already completed. Students might decide to do this when they are unsure if they have completed a feature correctly and want to see an example for confirmation.

2.3 Expert Evaluation

To test the feasibility of our method before building the whole system and conducting a user study, we did a preliminary expert evaluation of our algorithm. We generated example pairs to support student code snapshots from our historical dataset and evaluated their quality. To simulate real student help requests where an example might be needed, we selected snapshots that corresponded to times when historical students requested hints from iSnap. As in prior work on evaluating feedback [22], we sampled up to two hint requests (and their corresponding student code snapshots) from each student. We sampled 50 hint requests in total including 20 in F16, 20 in S17, and 10 in F17. For each hint request we generated four example code pairs using different techniques: DDF, DDF-IS, and our two baselines, Student and

Expert (explained in the next section). For the examples derived from student data (DDF, DDF-IS, Student), we generated examples using semester-based 3-fold cross-validation. For each semester, we used the other two semesters’ data as training data to generate the examples. We derived 9, 9, and 10 data-driven features for F16, S17, and F17, respectively.

Two co-authors, who neither authored examples nor worked on the algorithm itself, served as experts to evaluate the generated example pairs. Both experts have extensive experience in Snap! and the Squirrel assignment. We built an interface in iSnap to present each expert with the student’s original code and the example pair. Then we asked the experts to assess the example pair based on a detailed example code rating rubric², adapted from [22]. Our rubric has 4 attributes, each rated 1, 2 or 3, with higher scores being better. These 4 attributes measured: 1) **Relevance**: how relevant the suggested example code pair is to the student’s current goals, 2) **Progress**: how well the example code pair helps students make progress towards the final correct solution, 3) **Appropriateness & Interpretability**: how likely a tutor will be to suggest this example pair to a student and how easily a novice could understand the intention of the suggested example pair and 4) **Similarity**, how similar is the “start code” to the student’s code. The first 3 attributes are meant to assess the quality of the example. The 4th is meant to help us understand the relationship between example similarity and quality, since all examples were selected based on their similarity to student code. During evaluation, experts had access to students’ code history, and based their ratings on the student’s individual context.

To ensure that the two experts had a similar understanding of the rubric, they rated 10 examples together, which were not used in this study. They then rated the 200 examples pairs used in this study in 2 rounds of 100 each³. In Round 1, they independently rated 40 example pairs and then discussed their ratings to resolve any conflicts and reach consensus. Their inter-agreement reliability across the 40 example pairs achieved squared-weighted Cohen’s kappas of 0.94, 0.91, 0.82, 0.90 for Relevance, Progress, Appropriateness & Interpretability, and Similarity, respectively, indicating very strong agreement. They then split the remaining 60 pairs and rated them individually⁴. In Round 2, one expert rated the remaining 100 pairs individually, and the other expert rated 75 of these, which were discussed until consensus was reached. Across the 115 example pairs that were rated by both experts, the total squared-weighted Cohen’s kappa was 0.77.

2.3.1 Baselines

We compare both the interactive and non-interactive data-driven example-based feedback against two baselines: Expert-authored examples and examples generated naively from Student data. For data-driven example-based feedback, we use the two strategies (DDF and DDF-IS) described in section 2.2.4. Our goal for the **Expert baseline** was to reflect

²Available at: <http://go.ncsu.edu/edm2019-rubric>

³Due to the order in which examples were generated, Round 1 included DDF and Expert examples, and Round 2 contained Student and DDF-IS examples. However, raters were blind to the condition of the example.

⁴The rated examples were in DDF and Expert conditions.

a straightforward way of generating example-based feedback using a small number of expert-authored example pairs, which an instructor might reasonably create. This corresponds to the Next Step of the Nearest Sample Solution (NSNSS) strategy introduced by Gross et al. [9], which they found to be optimal. This strategy selects the next step of the nearest expert solution. To generate expert example pairs, two experts (specifically, two co-authors who did not rate the example pairs during evaluation) manually authored example pairs for *all* steps in the most common solution paths, which at least 10% of students took to solve the problem. Each of the two co-authors created the example code separately, based on the understanding that the example code should be useful to students and no extra explanations will be used to help students understand the suggested example code. During this process, the experts could review as many students' code (including all the history) as they needed. Afterwards, they met and discussed all the example pairs that they authored and came to consensus on the example pairs. In the Squirrel assignment, the common solution graph has 14 nodes and 13 edges, of which 7 were on a primary solution path and 6 were on two alternative paths. To select an Expert example-pair for a student, we first used the SourceCheck algorithm to sort the Expert example pairs based on the size of the "start code" and the similarity between their "start code" and the student's current code. We select the example pair whose "start code" accomplishes fewer features than the student's code and is very similar to the student's code.

It may seem unfair to compare the Interactive Data-driven examples to (non-interactive) Expert examples. However, we note that it would not be reasonable to create an *Interactive*-Expert baseline. Since the Expert examples consist of a small number of hand-authored examples, which completed features in a specific order (e.g. Feature 1 is *always* completed before Feature 2), it is not reasonable to give a student the option of selecting a desired example. If the student has only completed 2 features, they could simply select the example for the final feature and see a full solution. Since the goal of example-based feedback is to show only a single, incomplete feature, we always selected the closest Expert example-pair to the student's current code. By contrast, our data-driven approach generates enough example-pairs that we are able to provide many choices of features to complete, without revealing any other features in the process.

Our goal for the **Student baseline** was to reflect a naive approach to extracting examples from student code, without the feature-based cleaning and selection of our own algorithm. Gross et al. [9] defined their baseline of student-derived examples to show only students' submitted solutions, essentially giving away the whole answer. Instead, our baseline extracts multiple examples from students, based on when they ran their code. We hypothesized that students often run their code when they have completed a meaningful feature, making this a meaningful way to demarcate examples. We extracted examples from all correct student solution traces that did not request help. We selected student code snapshots based on when they ran their code. We treated consecutive run events within 15 seconds of one another as a single run event, and we took the last of these events as the boundary between example pairs. For each

consecutive pair of run events (more than 15 seconds apart), we extract an example pair consisting of the two corresponding snapshots. The code snapshot that happened earlier serves as the "start code" of the example pair, and the one happened later serves as the "end code". When selecting an example pair to show as feedback for a given student, we used the SourceCheck algorithm to find the nearest "start code" and present that example pair as feedback.

3. RESULTS

We structured our analysis around the following research questions: **RQ1**: Can we create useful example-based feedback naively from student data, without a data-driven algorithm? **RQ2**: How does the quality of data-driven example-based feedback compare with that of expert-authored example-based feedback? **RQ3**: Can the quality of data-driven example-based feedback be improved if an example is selected interactively, rather than automatically?

We address each RQ by comparing the quality of example-based feedback generated by four feedback approaches explained above: 1) Expert-authored (Expert), 2) Naive Student Data (Student), 3) Data-Driven Features (DDF), and 4) Data-Driven Features with Interactive Selection (DDF-IS). We evaluated quality in terms of Relevance, Progress, and Interpretability, as explained above. For RQ1, we hypothesized that our results would be consistent with Gross et al. [9] that naively extracted student examples would not lead to high-quality feedback. For RQ2, we hypothesized that data-driven example-based feedback would be more relevant to the student's code than expert feedback and just as useful otherwise. For RQ3, we hypothesized that interactive selection would improve the quality of data-driven example-based feedback.

3.1 Feature Coverage

From 106 student solution traces, the DDF algorithm was able to generate 13,927 unique data-driven examples, as shown in Table 2. Of these, only 728 (5%) were derived directly from a student's trace, and the rest were generated with the recursive algorithm explained in Section 2.2.3. It took around 10 minutes (645 seconds) to generate all the data-driven example pairs. Table 2 shows the number of examples generated by each algorithm. It also gives the "snapshot coverage" and "hint request coverage" of each algorithm. The former refers to the percent of all observed snapshots which had an available example in the same feature-state (meaning the example started with the same set of features completed). For this calculation, we used the expert-authored features defined in [34]. Hint request coverage considers only the 50 hint request snapshots we evaluated. These numbers are averaged across the 3 semesters.

Table 2: Total number of generated example pairs, corresponding average snapshot coverage and hint coverage in the expert-defined feature space.

Algorithm	DDF & DDF-IS	Stud.	Exp.
# of examples generated	13,927	242	13
Snapshot coverage	0.843	0.670	0.709
Hint request coverage	0.821	0.709	0.687

3.2 Expert Ratings

Our dataset consists of 50 hint requests and 200 example pairs (one example pair per feedback approach for each hint request). All the example pairs were rated on four attributes on a scale of 1-3: Relevance, Progress, Appropriateness & Interpretability, and Similarity, as described in Section 2.3. We found the first three attribute ratings showed significant positive pairwise Spearman correlations ranging from 0.51 to 0.81 (all $p < 0.001$). Similarity also had a lower, positive correlation with the other attributes, ranging from 0.24 to 0.34 (all $p < 0.001$). Due to the high positive correlation of the Relevance, Progress, and Appropriateness & Interpretability attributes, we also compute a Quality attribute, which sums all three attributes, with scores ranging from 3 to 9. Because all 4 example-based feedback approaches selected examples using SourceCheck’s code similarity function, we also investigated the relationship between SourceCheck’s calculated similarity and the expert-rated Similarity. We found a significant, positive correlation ($\rho = 0.29; p < 0.001$), suggesting that the similarity function is reasonable but could be improved. Table 3 reports mean values of each attribute for each example-based feedback approach⁵.

Table 3: Mean attribute ratings (with standard deviation) for example pairs in from each approach.

N = 50	Relev.	Prog.	A.&I.	Qual.	Simil.
DDF-IS	2.62 (0.73)	2.46 (0.76)	2.22 (0.82)	7.30 (2.05)	2.22 (0.86)
Expert	2.24 (0.89)	2.36 (0.80)	2.14 (0.90)	6.74 (2.24)	2.18 (0.80)
DDF	2.12 (0.92)	2.06 (0.89)	1.84 (0.82)	6.02 (2.38)	2.28 (0.88)
Student	1.72 (0.90)	2.12 (0.90)	1.80 (0.86)	5.64 (2.20)	2.24 (0.87)

To address our research questions, for each attribute we used Kruskal-Wallis test to determine if there was a significant difference in ratings across feedback generation approaches. For the overall Quality attribute, we found a significant difference among conditions ($\chi^2(3) = 16.06, p < 0.001$). We performed a post hoc Dunn’s test with Benjamini-Hochberg correction for multiple comparisons⁶ [2, 5] to identify pairwise significant differences between approaches. This showed a significant difference between Expert and Student examples ($z = 2.48, p = 0.026, r = 0.25^7$), DDF and DDF-IS ($z = 2.76, p = 0.017, r = 0.28$), DDF-IS and Student ($z = 3.69, p = 0.0013, r = 0.37$), suggesting that for overall quality, Student < DDF < DDF-IS, and Student < Expert.

We then inspected the difference for each individual attribute. A Kruskal-Wallis test showed a significant difference among approaches for the Relevance ($\chi^2(3) = 24.45, p < 0.001$). A post-hoc test using Dunn’s test with Benjamini-Hochberg correction showed a significant difference between

DDF and Student ($z = 2.14, p = 0.049, r = 0.21$), Expert and Student ($z = 2.79, p = 0.016, r = 0.28$), DDF-IS and DDF ($z = 2.76, p = 0.011, r = 0.28$), DDF-IS and Student ($z = 4.90, p < 0.001, r = 0.48$), DDF-IS and Expert ($z = 2.11, p = 0.04, r = 0.21$), but we did not find a significant difference between DDF and Expert ($z = 0.65, p = 0.515, r = 0.07$). This suggests that for Relevance, Student < DDF = Expert < DDF-IS.

We also found a significant difference among conditions for the Appropriateness & Interpretability ($\chi^2(3) = 8.98, p = 0.030$). However, a post-hoc test using Dunn’s test with Benjamini-Hochberg correction did not find any pairwise significant differences between conditions. For the other two attributes, similarly, we did not find any significant difference between conditions: Progress ($\chi^2(3) = 7.14, p = 0.068$); Similarity ($\chi^2(3) = 0.62, p = 0.89$). The results suggest that DDF-IS example pairs are more relevant to student code and are equally helpful and interpretable compared with expert-authored examples.

3.3 Inspection of Examples

To better understand our quantitative results, we manually inspected examples generated by each approach to understand how they differed and how those differences impacted the expert ratings. We investigated the following questions and present situations that highlight possible answers:

Why does the naive Student baseline have low quality? Our quantitative results show that our naive Student baseline does not produce useful example pairs. We manually investigated some Student example pairs to better understand why. Recall that the Student baseline extracts examples showing the changes between consecutive run of student code. We derived 242 of these example pairs across 3 semesters from 39 correct student submissions that did not request hints. Upon inspection, it is clear that this segmentation approach did not always produce meaningful example pairs. For example, a Student example pair might simply rearrange the order of some code blocks. This is a common debugging behavior, but it does not usually yield a useful example. We also found that even when Student example pairs demonstrated a meaningful step, they were often selected for students who had already completed that feature. This is not surprising, since the Student baseline selected the example with the most similar start code, but this does not guarantee the student will not have additional features completed. Lastly, many Student examples contained extraneous code that made them harder to interpret. These three problems – lack of meaningful steps, repeating completed steps, and extraneous code – are all addressed in our DDF-based example selection. With good features, we can identify meaningful example steps, ensure that all examples complete new features, and remove extraneous code.

When were data-driven examples more adaptive than Expert examples? Our quantitative results suggest that the DDF-IS approach was able to generate examples that were significantly more Relevant than those of the Expert approach. We hypothesized that this was enabled by the large number of unique example pairs generated by DDF (13,927). By contrast the Expert approach used only 13 examples pairs, which covered the common solution paths

⁵Although attribute ratings are ordinal, we report the mean and SD, since the median values for each attribute are generally the same (2 or 3)

⁶We report p-values corrected with the Benjamini-Hochberg procedure to control the false discovery rate at 0.05, keeping the α significance threshold at 0.05.

⁷The effect size r is calculated as described in [25]

Student Code	DDF-IS	Expert
<pre> Squirrel(rotations, length): Clear() PenDown() Repeat(4 * rotations) Move(10) Turn(90) Change(length)by(25) </pre>	<pre> Squirrel(rotations, length): PenDown() Repeat(4 * rotations) Move(10) Move(rotations) Turn(90) Change(length)by(5) </pre>	<pre> - DrawSquirrel(): + DrawSquirrel(rotations): PenDown() Repeat(4) Repeat(rotations) Move(10) Turn(90) PenUp() </pre>

Figure 3: DDF-IS and Expert examples for one snapshot (red/green indicate changed code in the example).

that at least 10% of students took to solve the problem. While some of DDF-IS’s improvement may have come from its Interactive Selection step (explored below), our results in Section 3.1 also show DDF examples have larger coverage than the expert-authored examples for both the hint requests and student snapshots. Our manual inspection uncovered a number of situations when a relevant expert Example was not available, but a DDF example was. For example, a strong minority of students solved the Squirrel problem by using a second input (parameter) to store the side length of shape. Figure 3 shows that for the same hint request, the DDF-IS can suggest examples not only similar to the student code, but also relevant to what the student is working on. However, the expert-authored example only has one parameter in the custom block and suggests something that the student has already done. The Expert example becomes less helpful and irrelevant when a student has solution that deviates from the Expert examples. However, our data-driven examples are more adaptive in this scenario and can provide examples both similar and relevant to the student. We note that this adaptivity was enabled in large part by the recursive example generation algorithm outlined in Section 2.2.3. Only 2 of the 50 examples selected by DDF-IS were extracted directly from student code traces (the “naive approach”); the other 48 were generated synthetically.

When did DDF-IS select better examples than DDF?

Since DDF and DDF-IS were selecting examples from the same pool of examples, but DDF-IS has significantly higher overall quality, we wanted to understand where the selection algorithms differed. We investigated some pairs and focused on when our algorithm failed to select relevant examples. In those scenarios, we found that the DDF would suggest the student to add a ‘pen down’, or add a custom block in the main script area. Those features are necessary for a correct solution, but they may have lower priority compared with other features such as “control the sprite to draw a square” or “repeat the spiral the correct number of times.” Additionally, we also found that the data-driven feature detector sometimes counted a feature as complete when it still had a bug or a missing block. Thus, it would suggest example pairs that accomplish new features without fixing the broken one. In the opposite case, the student may have already finished a feature, but the feature detector sometimes failed to detect it, showing a redundant example. This occurred frequently when students accomplished a feature in a unique way, not captured by our data-driven feature definition. For example, a few students attempted to use a recursive approach to solve Squirrel, but since this was quite uncommon, our data-driven features failed to generate meaningful example pairs for this solution. The DDF-IS can help resolve the first two cases, since students can choose whether they

need help on a feature that the DDF failed to detect.

4. DISCUSSION

RQ1: Can we create useful example-based feedback naively from student data, without a data-driven algorithm? Our results suggest that naively extracting examples from student code based on when students ran their code does not yield high-quality example-pairs. These naive Student examples were lowest or second-lowest, rated on every dimension, with significantly lower Relevance than all other approaches and significantly lower overall Quality than the DDF-IS approach and the Expert-authored examples. This is consistent with our hypothesis for RQ1. It is also in agreement with the work of Gross et al. [9], who found that using student solutions as example-based feedback were rated lower by experts and led to less improvement in students than other, expert-authored examples. However, rather than using a student’s submitted code as an example as Gross et al. did, we used the changes that students made between running their code. We believe that this represents a more reasonable student baseline, since it only shows a part of the answer like the other feedback approaches. It should also theoretically limit the cognitive load needed for students to learn from the examples. Our results show that even so, a naive approach does not create high-quality student examples, compared to other approaches. Our data-driven approach attempts to address this by creating examples based on when students completed features.

One of the goals of using student data to generate examples is that they can more closely match the student’s current code. In fact, we were able to extract 242 unique Student examples, compared with 13 for the Experts. This did enable our system to identify Student examples that were more *Similar* to the help-requesting student’s code than Expert examples (though not significantly more). However, it is also clear that Similarity alone did not translate into Relevance, since the Student examples had the lowest Relevance scores. We cannot guarantee that students make meaningful changes between two consecutive snapshots, and thus the generated example pairs may not accomplish a meaningful chunk of code. For example, the Student example pair may suggest something that a student has already done. In this case, we could use the data-driven features to identify a later snapshot with more relevant changes as the “end code.” This suggests the need for a more deliberate, data-driven process that can still create a large number of examples to select from without sacrificing quality.

RQ2: How does the quality of data-driven example-based feedback compare with that of expert-authored example-based feedback?

We hypothesized that our Data-Driven Features (DDF) examples would be more relevant to help-requesting student’s code than the expert baseline, with similar levels of Progress and Interpretability. However, our results did not support this hypothesis. There were no significant differences between the two approaches, with the DDF algorithm performing similarly on Relevance and slightly worse on Progress and Interpretability. Our original hypothesis was based on the premise that data-driven examples would be more Relevant, since they are selected from a much larger database of examples (4,000 to 5,000) that represents a large variety of student code. While this was

not confirmed, our results do suggest that our current, *non-interactive* data-driven algorithm performs significantly better than naive Student examples and only marginally worse than Expert examples. Since it is often difficult to get instructors to author examples [12], this suggests our DDF examples should be a reasonable substitution.

There are a few possible explanations for the poorer performance of DDF. First, it is possible that Squirrel problem we analyzed may not have been complex enough to necessitate generating a large variety of examples, which is one of the key proposed advantages of our algorithm. If most hint-requesting students performed the steps of the problem in the same order as the Expert examples, these Expert examples would generally be Relevant, leaving less room for the DDF examples to improve upon them. We have some support for this explanation, since across semesters 53.8% to 85.7% of the hint requests we analyzed had an expert feature state that matched one of our Expert examples exactly. However, we also note that the expert baseline was rated less than 3 for Relevance 46% of the time, so there was clearly room for improvement, as we discuss later.

It is also clear that the DDF approach did not perform as well as expected, scoring between the Expert and Student baselines for Progress and Interpretability. There are two possible explanations for this: either the algorithm is failing to *generate* good example pairs, or it is generating useful example pairs, but failing to *select* the best pair to show a given student. The higher scores of the DDF-IS algorithm, which generated the same example pairs as DDF, but allowed a human to select the best one, suggest that the problem lies in the DDF algorithm's selection of example pairs. This selection process (explained in Section 2.2.4) primarily uses the SourceCheck algorithm to identify the example with the most similar start state to the student's code. While this approach did lead to DDF having the highest Similarity ratings, there was only a $\rho = 0.238$ correlation between Relevance and Similarity. This suggests that Similarity alone is not a good proxy for example Relevance or overall quality. The challenge of automatically selecting an appropriate example also reflects prior work on data-driven feedback [23], where low-quality hints arose because the hint generation algorithm was unable to identify the most useful hints and filter out less useful hints.

RQ3: Can the quality of data-driven example-based feedback be improved if an example is selected interactively, rather than automatically? Our results show that the quality of data-driven example-based feedback can be improved if the examples are selected interactively. The DDF-IS algorithm scored significantly higher than the DDF algorithm on Quality and Relevance, even significantly outperforming the expert baseline on Relevance, with the best overall scores. Recall that for the DDF-IS examples, we manually tagged the data-driven features with natural language labels, which a student could reasonably select from when making an example request. In this preliminary evaluation, we simulated this selection process by manually selecting the best *data-driven feature* to show to each student. Importantly, we only selected the feature (e.g. “move the sprite using a variable”), not the example, and the algorithm still selected the best possible example for a given feature

(from hundreds of possible choices). Still, it is likely that an expert familiar with the system is more likely to choose an appropriate feature than a student, so this represents an *upper bound* for how well an interactive, data-driven example selection algorithm could reasonably perform. These results show that, when this proper selection is applied, the larger database of example pairs generated by our algorithm can lead to more Relevant examples than a static set of expert-authored examples. This represents a growing trend among data-driven systems that support programming to leverage human (i.e. student and instructor) knowledge in conjunction with data-driven algorithms [10, 18, 33]. However, from our preliminary expert evaluation, we can not make claims about how students would actually react to such a choice. Our results also suggest ways that we can improve the *automated* example selection procedure. The majority of low-quality DDF example pairs would select a feature that the student needs but could be less relevant than other features. For example, one example pair suggests adding the student's custom block into the main script so they can test it. However, in Snap!, students can click on the custom block to test it directly. This feature should therefore have lower priority than others. For the example shown in the Results section, the data-driven feature detector failed to detect correct variations in student code, and suggested a feature that the student had already completed. These examples suggest that it may be more important to select the most important feature to demonstrate, before considering the similarity of the example to the student's current code.

5. CONCLUSION

This study presents a method to generate data-driven examples automatically from historical student data. We evaluated two versions of data-driven generated examples using the student code when they need help, compared with two baselines: examples authored by experts and examples derived naively from student solution traces. Experts evaluated all of the example pairs based on a multidimensional rubric. Our preliminary results demonstrate that by selecting appropriate features, our data-driven examples can be more relevant than both baselines while retaining the usefulness and interpretability. These promising results suggest that our method can generate adaptive, data-driven examples automatically with quality similar to that of expert-authored examples. Even though our method is based on a block-based programming environment, it is language-agnostic and may also be generalized to textual programming languages; though further studies are needed to verify this proposed generalizability.

Limitations: This preliminary study relied on experts ratings to assess example quality, yielding valuable insight, but future work is needed to determine whether these results translate into improved student performance and learning. For example, even a well-rated example pair may still give away too much of the answer and impede learning. We also do not know how our results will generalize beyond the single assignment evaluated here, since we only applied this method to a short, block-based problem with a solution that is usually 7-10 lines of code. In our DDF-IS condition, we had an expert interactively select the feature to be presented, rather than a student. As noted earlier, this is likely an optimistic implementation, as we do not know

whether students can effectively select examples. Finally, all conditions used the SourceCheck algorithm to select similar examples to the student's code, but other approaches may better capture example relevance (e.g. [11]). We are currently planning a student evaluation of the DDF algorithm to address these limitations.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under grant 1623470.

7. REFERENCES

- [1] N. Augsten, M. Böhlen, and J. Gamper. The pq-gram distance between ordered labeled trees. *TODS*, 2010.
- [2] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society*, 57(1):289–300, 1995.
- [3] M. E. Caspersen and J. Bennedsen. Instructional design of a programming course: a learning theoretic approach. In *ICER (Workshop)*, pages 111–122, 2007.
- [4] J. Coenen, S. Gross, and N. Pinkwart. Comparison of feedback strategies for supporting programming learning in integrated development environments (ides). In *ICCSAMA*, pages 72–83. Springer, 2017.
- [5] O. J. Dunn. Multiple comparisons using rank sums. *Technometrics*, 6(3):241–252, 1964.
- [6] B. J. Ericson, J. D. Foley, and J. Rick. Evaluating the efficiency and effectiveness of adaptive parsons problems. In *ICER*, pages 60–68. ACM, 2018.
- [7] B. J. Ericson, L. E. Margulieux, and J. Rick. Solving parsons problems versus fixing and writing code. In *Koli Calling*, pages 20–29, 2017.
- [8] P. Gerjets, K. Scheiter, and R. Catrambone. Designing instructional examples to reduce intrinsic cognitive load: Molar versus modular presentation of solution procedures. *Instructional Science*, 32(1-2):33–58, 2004.
- [9] S. Gross, B. Mokbel, B. Hammer, and N. Pinkwart. How to select an example? a comparison of selection strategies in example-based learning. In *ITS*, pages 340–347. Springer, 2014.
- [10] A. Head, E. Glassman, G. Soares, R. Suzuki, L. Figueredo, L. D'Antoni, and B. Hartmann. Writing Reusable Code Feedback at Scale with Mixed-Initiative Program Synthesis. In *L@S*, 2017.
- [11] R. Hosseini and P. Brusilovsky. A study of concept-based similarity approaches for recommending program examples. *New Review of Hypermedia and Multimedia*, 23(3):161–188, 2017.
- [12] I.-H. Hsiao and P. Brusilovsky. The role of community feedback in the student example authoring process: An evaluation of annotex. *British Journal of Educational Technology*, 42(3):482–499, 2011.
- [13] M. Ichinco, K. J. Harms, and C. Kelleher. Towards understanding successful novice example user in blocks-based programming. *Journal of Visual Languages and Sentient Systems*, 3:101–118, 2017.
- [14] M. Ichinco and C. Kelleher. Exploring novice programmer example use. In *VL/HCC*, 2015.
- [15] H. Keuning, J. Jeuring, and B. Heeren. A systematic literature review of automated feedback generation for programming exercises. *TOCE*, 19(1):3, 2018.
- [16] B. B. Morrison, L. E. Margulieux, and M. Guzdial. Subgoals, context, and worked examples in learning computing problem solving. In *ICER*, 2015.
- [17] E. Patitsas, M. Craig, and S. Easterbrook. Comparing and contrasting different algorithms leads to increased student learning. In *ICER*, pages 145–152, 2013.
- [18] C. Piech, J. Huang, A. Nguyen, M. Phulsuksombati, M. Sahami, and L. Guibas. Learning program embeddings to propagate feedback on student code. In *ICML*, pages 1093–1102, 2015.
- [19] T. W. Price, Y. Dong, and T. Barnes. Generating data-driven hints for open-ended programming. *EDM*, 16:191–198, 2016.
- [20] T. W. Price, Y. Dong, and D. Lipovac. isnap: towards intelligent tutoring in novice programming environments. In *SIGCSE*, pages 483–488. ACM, 2017.
- [21] T. W. Price, R. Zhi, and T. Barnes. Evaluation of a data-driven feedback algorithm for open-ended programming. In *EDM*, 2017.
- [22] T. W. Price, R. Zhi, and T. Barnes. Hint generation under uncertainty: The effect of hint quality on help-seeking behavior. In *AIED*, 2017.
- [23] T. W. Price, R. Zhi, Y. Dong, N. Lytle, and T. Barnes. The impact of data quantity and source on the quality of data-driven hints for programming. In *AIED*, pages 476–490. Springer, 2018.
- [24] K. Rivers and K. R. Koedinger. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *IJAIED*, 2017.
- [25] J. Robertson and M. Kaptein. *Modern statistical methods for HCI*. Springer, 2016.
- [26] D. L. Schwartz, C. C. Chase, M. A. Oppezzo, and D. B. Chin. Practicing versus inventing with contrasting cases: The effects of telling first on learning and transfer. *Journal of Educational Psychology*, 103(4):759, 2011.
- [27] J. Sweller. The worked example effect and human cognition. *Learning and instruction*, 2006.
- [28] J. Sweller and G. A. Cooper. The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and instruction*, 1985.
- [29] J. Sweller, J. J. Van Merriënboer, and F. G. Paas. Cognitive architecture and instructional design. *Educational psychology review*, 10(3), 1998.
- [30] J. G. Trafton and B. J. Reiser. Studying examples and solving problems: Contributions to skill acquisition. In *CSS*, pages 1017–1022, 1993.
- [31] A. Vihavainen, M. Paksula, and M. Luukkainen. Extreme apprenticeship method in teaching programming for beginners. In *SIGCSE*, 2011.
- [32] L. Vygotsky. Interaction between learning and development. *Readings on the development of children*, 23(3):34–41, 1978.
- [33] M. Wu, M. Mosse, N. Goodman, and C. Piech. Zero shot learning for code education: Rubric sampling with deep learning inference. *arXiv preprint arXiv:1809.01357*, 2018.
- [34] R. Zhi, T. W. Price, N. Lytle, Y. Dong, and T. Barnes. Reducing the state space of programming problems through data-driven feature detection. In *EDM (Workshops)*, 2018.

A Multivariate Elo-based Learner Model for Adaptive Educational Systems

Solmaz Abdi
The University of Queensland
solmaz.abdi@uq.edu.au

Hassan Khosravi
The University of Queensland
h.khosravi@uq.edu.au

Shazia Sadiq
The University of Queensland
shazia@itee.uq.edu.au

Dragan Gasevic
Monash University
dragan.gasevic@monash.edu

ABSTRACT

The Elo rating system has been recognised as an effective method for modelling students and items within adaptive educational systems. The existing Elo-based models have the limiting assumption that items are only tagged with a single concept and are mainly studied in the context of adaptive testing systems. In this paper, we introduce a multivariate Elo-based learner model that is suitable for the domains where learning items can be tagged with multiple concepts, and investigate its fit in the context of adaptive learning. To evaluate the model, we first compare the predictive performance of the proposed model against the standard Elo-based model using synthetic and public data sets. Our results from this study indicate that our proposed model has superior predictive performance compared to the standard Elo-based model, but the difference is rather small. We then investigate the fit of the proposed multivariate Elo-based model by integrating it into an adaptive learning system which incorporates the principles of open learner models (OLMs). The results from this study suggest that the availability of additional parameters derived from multivariate Elo-based models have two further advantages: guiding adaptive behaviour for the system and providing additional insight for students and instructors.

1. INTRODUCTION

Adaptive educational systems make use of data about students, learning process, and learning products to adapt the level or type of instruction for each student. Commonly, this adaptivity takes the form of selecting items from a large repository of learning resources to match the current learning ability of a student [17]. To do so, adaptive educational systems rely on learner models that capture an abstract representation of a student's ability level based on their performance and interactions with the system [6]. Two conventional approaches have been heavily studied for modelling learners. (1) Bayesian Knowledge Tracing (BKT) uses

a Hidden Markov Model for capturing students' knowledge state as a set of binary variables indicating whether a knowledge component has been mastered [4]. (2) Item Response Theory (IRT) [11] and its extensions such as Additive Factor Model (AFM) [3] and Performance Factor Analysis (PFA) [18] rely on a logistic regression model to estimate latent traits related to students' knowledge state and the difficulty of learning items. Neither of these approaches, however, can be easily integrated into online adaptive educational systems as they generally require pre-calibration on big samples of data and ongoing addition of new students and new learning items to the system necessitates continuous calibration of model parameters [19].

The Elo rating system has been shown to be an effective alternative to the above mentioned conventional approaches for modelling students in adaptive educational systems [14]. It is simple, fast, robust and order-sensitive which makes it a suitable model for adaptive educational systems where it is required to update students' proficiency level upon administration of each question [23]. In the educational context, the Elo-based model is employed to conduct a paired comparison among students and learning items as competitors [23]. This model is self-correcting, meaning that the ratings, in the long run, should correctly reflect students' knowledge states and difficulty levels of questions [20].

The majority of the existing studies on Elo-based models have the following two characteristics: They (1) use repositories that contain items that are pure [9] and are tagged with a single concept and (2) are studied in the context of adaptive testing systems such as computerised adaptive testing (CAT) [5]. The contribution of this paper lies in (1) introducing a new variant of the Elo-based algorithm called M-Elo that has the capacity to model students and items using repositories that contain items that are tagged with one or more concepts, and (2) investigating its fit in the context of adaptive learning systems, which in contrast to adaptive testing systems, take a more student-centred approach with the aim of assisting students in their learning.

To evaluate the applicability of M-Elo in the context of adaptive learning where learning items are explicitly tagged with one or more concepts, we first compare the predictive performance of M-Elo against the standard Elo-based model using simulated data sets. The results from these experiments demonstrate that the behaviour of the M-Elo is consistent

Solmaz Abdi, Hassan Khosravi, Shazia Sadiq and Dragan Gasevic "A Multivariate ELO-based Learner Model for Adaptive Educational Systems" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 228 - 233

with expectations over a range of parameter settings, and therefore this provides some evidence to suggest that it will be robust in a real-world setting. We then compare the predictive performance of M-Elo against the standard Elo-based model using public data sets. The results from the experiments indicate that M-Elo have superior predictive performance compared to the standard Elo-based model, but the difference is rather small. Finally, we integrate M-Elo into an adaptive learning system and conduct a case study by piloting this system in a large introductory course on relational databases at The University of Queensland.

The results obtained from our case study suggests that using M-Elo, as a multivariate Elo-based model, in adaptive learning systems have two additional advantages beyond the standard Elo-based model: The first advantage lies in the availability of additional parameters that provide insight into the characteristics of the domain and the learning process which in turn can be used for guiding adaptivity. The second advantage is that when M-Elo is opened based on the concepts of open learner models (OLMs) [2], it can provide additional insight on course level and individual level competencies and gaps, that can be used by instructors for improving learning item design, while also providing meta-cognitive benefits for students, such as increased motivation and trust in the system. The conducted case study also revealed some of the shortcomings of using Elo-based models in the context of adaptive learning. The main issue is that Elo-based models consider items and students as identical rivals. This assumption seems to fail to adequately model the long-term use of an adaptive learning system in which students' knowledge may increase over time while the difficulty level of the items remains constant.

2. BACKGROUND

Elo-Based Learner Models. The Elo rating system is originally developed to rate chess players and is established based on paired comparison of data where two chess players compete against each other [23]. In the educational setting, a similar paired comparison can be conducted between a student and a question being attempted by the student. The standard implementation of the Elo-based model in education resembles the Rasch model employed in IRT that models students and questions with a single global parameter [20]. An important extension to the Elo-based model in education is the multivariate Elo-based model proposed by [9] in the context of psychometrics, where instead of using a global knowledge parameter for students, it uses an overlay model which estimates the competency of a student in each different concept using a separate parameter. A similar model for adaptive practice of facts was later proposed by [20], which had an additional global parameter compared to the multivariate Elo-based model that was used in combination with the concept level parameters in modelling the ability of a student on each concept. Both of these models make the assumption that items are tagged with a single concept, which limits their applicability in domains where items can be tagged with multiple concepts. For example, in a Programming domain, an arbitrary learning item might be associated with both "lists" and "loops" concepts. M-Elo, our proposed model, is an extension over the multivariate Elo-based model that has the capacity to model students and items in the presence of items with multiple concepts.

Adaptive Testing vs. Adaptive Learning. Adaptivity in educational systems has been investigated broadly both in the context of computerized adaptive testing (CAT) and adaptive learning [17]. Generally, adaptive testing systems conduct an exam using a sequence of questions that are successively administered with the purpose of maximising the precision of the system's current estimate of the student's ability. The exam is usually terminated once the system has an estimate of the student's ability with a confidence level that exceeds a user-specified threshold. In contrast, adaptive learning systems such as ALEKS¹ take a more student-centred approach with the aim of assisting students in their learning. As such, in most adaptive learning systems, students (1) have the opportunity to decide whether or not to engage with the suggested learning items, (2) can spend, theoretically, an infinite amount of time on a learning item or on the system and (3) will receive rich feedback on their learning after engaging with each learning item. In this paper, we examine the fit of our proposed model, and more generally multivariate Elo-based models in the context of adaptive learning.

Open Learner Models. Open learner models (OLMs) are learner models that are externalised and made accessible to students or other stakeholders such as instructors, often through visualisation, as an important means of supporting learning [2]. They have been integrated into a variety of learning technologies such as learning dashboards [1], intelligent tutoring systems [21] and adaptive learning platforms [8] to help students and instructors in monitoring, reflecting and regulating learning [2]. In this paper, we investigate the benefits of opening Elo-based models, such as M-Elo in adaptive learning system based on the principles of the OLMs.

3. ELO-BASED LEARNER MODELLING

In this section, we first define a mathematical notation for describing the models. Section 3.1 then provides a review of the standard Elo-based model in the educational context. Finally, Section 3.2 introduces our proposed variation of the multivariate Elo-based model, which we call M-Elo.

In what follows, let $U_N = \{u_1 \dots u_N\}$ denote a set of students who are enrolled in a course on an adaptive educational system, where u_n refers to an arbitrary student. Each course consists of a set of concepts $\Delta_L = \{\delta_1 \dots \delta_L\}$, referred to as the domain model, where δ_i presents an arbitrary concept. In this work, the notion of a concept is based on taxonomies of knowledge components described by [16]. Let $Q_M = \{q_1 \dots q_M\}$ present the content model, denoting a repository of learning items that are available to students in a course in the adaptive learning system, where q_m refers to an arbitrary item. These learning items can be tagged with one or more knowledge components; $\Omega_{M \times L}$ is a two-dimensional array, where ω_{ml} is $1/g$ if item q_m is tagged with g knowledge components including knowledge component δ_l , and 0 otherwise. Let a two-dimensional array $A_{N \times M}$ keep track of students' attempts on the items, where $a_{nm} = 1$ indicates that student u_n has answered item q_m correctly, and $a_{nm} = 0$ indicates that student u_n has answered item q_m incorrectly.

¹<https://www.aleks.com/>

3.1 The Standard Elo-based Learner Model (Elo)

In the standard Elo-based learner model (Elo), both students and items are considered as identical rivals. Elo assumes one student parameter θ_n indicating u_n 's global proficiency level on the entire domain and one item parameter d_m presenting the difficulty level of item q_m . It uses a logistic function to estimate the probability of a correct answer by a student to a given item based on the difference between the student's global proficiency level and the item's difficulty. [20]:

$$P(a_{nm} = 1 | \theta_n, d_m) = \sigma(\theta_n - d_m) \quad (1)$$

To account for the guessing effect in the case of multiple choice items with c possible options, Formula 1 can be easily replaced with a shifted logistic regression using: $P(a_{nm} = 1 | \theta_n, d_m) = \frac{1}{c} + (1 - \frac{1}{c}) * \sigma(\theta_n - d_m)$. once u_n has attempted q_m , the knowledge state of u_n and difficulty level of q_m are simultaneously updated using the following formulas:

$$\theta_n := \theta_n + K(a_{nm} - P(a_{nm} = 1 | \theta_n, d_m)) \quad (2)$$

$$d_m := d_m + K(P(a_{nm} = 1 | \theta_n, d_m) - a_{nm}) \quad (3)$$

, where K is a constant value determining the sensitivity of the estimations based on the student's last attempt. The updates to the student's knowledge state (Formula 2) and the difficulty of the item (Formula 3) in Elo follows the principles of zero-sum game, in which the sum of gains (loss) to the student's knowledge state and the loss (gain) to the difficulty of the item after the student answers the item turns out to be zero. In most extensions of the Elo-based models, in order to get to a stable estimations for the student's knowledge state and item difficulty, K is replaced with an uncertainty function

$$U(n) = \frac{\gamma}{1 + \beta * n} \quad (4)$$

, where γ and β are constant hyper-parameters determining the starting value and slope of changes, respectively, and n indicates the number of prior updates on student's knowledge state or item difficulty [20].

3.2 Multi-Concept Multivariate Elo-based Learner model (M-Elo)

In contrast to Elo, where only one parameter is used to model a student's knowledge state on the entire domain, M-Elo uses independent parameters to model the student's knowledge state on each individual concept in the domain, and a global parameter for modelling each item. As in Elo, for each learning item q_m there is a global difficulty d_m approximating the difficulty level of the item. For students, let a two-dimensional array $\Lambda_{N \times M}$ represents a student's Elo-based learner model, where λ_{nl} represents student u_n 's knowledge state on concept δ_l , approximating the proficiency level of the student on that certain concept. To estimate the probability that student u_n answers an item q_m correctly, we first compute $\bar{\lambda}_{nm} = \sum_{l=1}^L \lambda_{nl} \times \omega_{ml}$, which estimates u_n 's average competency on concepts that are associated with q_m . We then compute the probability of u_n answering q_m correctly using:

$$P(a_{nm} = 1 | \bar{\lambda}_{nm}, d_m) = \sigma(\bar{\lambda}_{nm} - d_m) \quad (5)$$

After u_n answers q_m , the updated estimate of d_m is obtained using:

$$d_m := d_m + K(P(a_{nm} = 1 | \bar{\lambda}_{nm}, d_m) - a_{nm}) \quad (6)$$

where K can be replaced with an uncertainty function $U(n)$ presented in Formula 4. To update student u_n 's Elo ratings based on the given answer to item q_m , we update the student's parameter on each concept δ_l the question is tagged with separately using the following formula:

$$\lambda_{nl} := \lambda_{nl} + \alpha.K(a_{nm} - P(a_{nm} = 1 | \lambda_{nl}, d_m)) \quad (7)$$

where α is a normalisation factor, ensuring that the zero-sum game principles are enforced in the model. As such, the net change made to the parameters estimating u_n 's proficiency level in the concepts that are associated with q_m (computed by Formula 7) and d_m (computed by Formula 6) sum to zero. α is computed using the following formula:

$$\alpha = \frac{|P(a_{nm} = 1 | \bar{\lambda}_{nm}, d_m) - a_{nm}|}{\sum_{l=1}^L (|a_{nm} - P(a_{nm} = 1 | \lambda_{nl}, d_m)| \times \omega_{ml})} \quad (8)$$

4. EVALUATION

In this section, we evaluate the suitability of M-Elo in the context of adaptive learning systems. Section 4.1 compares the predictive performance of M-Elo against Elo using a suite of simulated data sets. Section 4.2 compares the predictive performance of M-Elo against Elo using publicly available data sets. As commonly used in the evaluation of learner models, we use the area under the curve (AUC), root mean squared error (RMSE) and accuracy (ACC) for reporting the predictive performance of the models. For all experiments, students' knowledge states and item difficulties in Elo and M-Elo are initialised to zero. Section 4.3 then reports the results of a case study that integrates M-Elo into an adaptive learning system which incorporates the principles of OLMs.

4.1 Synthetic Data Sets

Synthetic data sets were used to assess the behaviour of the models under different settings by varying parameters in the data generation template. The synthetic data sets were generated using a sequence of steps proposed by [13]. At first, a set of students with predefined knowledge states over a set of knowledge components were created. Assigning students' knowledge state was performed by sampling from a normal distribution, where the mean of distribution for each student was sampled from a uniform distribution. In this model, the standard deviation (σ) of the normal distribution determined the complexity of a student's knowledge state, where smaller values of σ led to having students that had roughly the same ability across all of the knowledge components and bigger values of σ led to having students with a higher diversity on their abilities across all of the knowledge components. Then, a set of learning items with pre-defined concepts, level of difficulty and discrimination was generated. Assigning concepts to items was performed by sampling from a discrete uniform distribution, while difficulty and discrimination were sampled from a normal distribution. Lastly, to compute the probability that a student u_n answered a learning item q_m correctly, a 2PL Item Response Theory (IRT) [10] model was used as recommended by [7] using: $\frac{1}{1 + e^{-a_m(\theta_n - b_m)}}$, where, θ_n represents a student's average competency on the concepts associated with

the item, and b_m and a_m were the difficulty level and the discrimination level of the item q_m , respectively. In all synthetic data sets, 100 students, 1000 learning item and 70,000 answers were sampled completely at random. Each learning item was tagged with one to three concepts. In these experiments, 70% of data was used for training and the remaining 30% was reserved for the test. Each experiment was repeated five times and the reported values are the average results across the five runs.

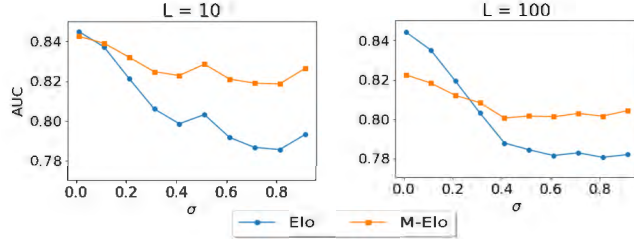


Figure 1: AUC as σ is increased. Results are reported for $L = 10$, and $L = 100$

Figure 1 compares the AUC performance of Elo and M-Elo in estimating students' knowledge states with respect to different values of σ under two different settings for the number of knowledge components (L). Our results for $L = 10$ suggest that for smaller values of σ , Elo outperforms M-Elo. This was predictable, as in this setting, each student has roughly the same competency level on all of the concepts, and since Elo relies on a global knowledge estimation for students proportional to their overall performance, it can outperform M-Elo in this setting. As σ is increased and students with more diversity in their abilities across different concepts are generated, M-Elo outperforms Elo as it is able to discriminate between students' abilities on each individual concept, leading to more accurate estimations of students' knowledge state compared to Elo. By increasing L to 100, the same trend is observable; however, the intersection point for σ where M-Elo outperforms Elo becomes bigger, as in this scenario, with the same amount of data, M-Elo needs to learn many more parameters independently. Evaluations using RMSE and ACC led to very similar patterns as AUC. Therefore, in the interest of space, figures reporting these results have not been included.

4.2 Public Data Sets

Three data sets namely 'Algebra I 2005-2006' (Alg2005), 'Algebra I 2006-2007' (Alg2006) and 'Bridge to Algebra 2006-2007' (BAI2006), which were obtained from the PSLC Datashop were used [15]. These data sets were originally obtained from Carnegie Learning's Cognitive Tutor and were made available as the "DevElopment" sets in KDD Cup 2010 [22]. Cognitive Tutor provides a fine representation of knowledge components associated with each item. It is a formative assessment tool, where each step taken by a student to answer a problem is considered as an individual interaction. Each learning item (referred to as 'Step' and presented by 'Step Name' in the data sets) is associated with one or more concept (KC) covered in the course. We used the train/test split provided by KDD Cup 2010 and discarded interactions with items that have not clearly been tagged with particular concepts. No students were discarded. Overall informa-

tion about these data sets are presented in Table 1. A grid search was conducted to determine optimal values for hyperparameters γ and β of the uncertainty function, described in Formula 4. As also reported by [20], the results were not really sensitive to changes from these parameters. In all the reported experiments on public data sets, γ was set to 1.8 and β was set to 0.05.

Table 1: Public data sets

Data Set	Students	KC	Items	multi-KC ²	Interactions
Alg2005	575	112	147,914	51,171	609,979
Alg2006	1840	714	319,151	21,415	1,825,030
BAI2006	1146	493	19,954	1,650	1,822,697

Table 2 compares the AUC, RMSE and accuracy (ACC) of the model fit statistics related to each model for estimating students' knowledge state. As it is indicated, on all three data sets M-Elo outperformed Elo in predicting student performance, but the difference was rather small. Considering the insights obtained from the experiments with synthetic data sets, where M-Elo was outperforming Elo with a small margin, it may be possible to hypothesise that students often have different competency levels on different concepts, but these differences are often not too significant.

Table 2: AUC, RMSE and ACC for public data sets

Data Set	AUC		RMSE		ACC	
	Elo	M-Elo	Elo	M-Elo	Elo	M-Elo
Alg2005	0.726	0.750	0.392	0.385	0.787	0.79
Alg2006	0.687	0.695	0.394	0.390	0.784	0.797
BAI2006	0.676	0.712	0.368	0.361	0.827	0.828

4.3 Case Study

To investigate the fit of M-Elo for adaptive learning in an authentic environment, we integrate M-Elo into an adaptive learning system called RiPPLE [12] and piloted the platform in an introductory course on relational databases at The University of Queensland. The course covers many concepts that are generally included in an introductory course on relational databases including conceptual database design using ER diagrams, relational models, functional dependencies, normalisation, relational algebra, Structured Query Language (SQL), data warehousing and database security. The platform was used for 13 weeks; during this period, 521 of the students enrolled in this course made 91,340 attempts on 1,632 learning items which were available in the system. Among these items, 144 items were tagged with two or more of the 17 concepts, which were associated with the course. Our aim was to investigate the benefits and shortcomings of using M-Elo in an adaptive system where the model is shared with the students based on the principles of OLMs through a visualisation widget, as indicated in figure 2, which allows students to visually see their current knowledge state on all concepts of the course. This visualisation is updated and represented to students as soon as they answer a new item from the item pool.

Guiding adaptivity. To guide the adaptivity of the system, the estimated knowledge states of the students and difficulties of the questions, as computed by M-Elo, are passed to the recommendation engine of the adaptive system. For each student, the recommendation engine recommends questions

²multi-KC in Table 1 indicates the number of items tagged with two or more knowledge components (KCs)

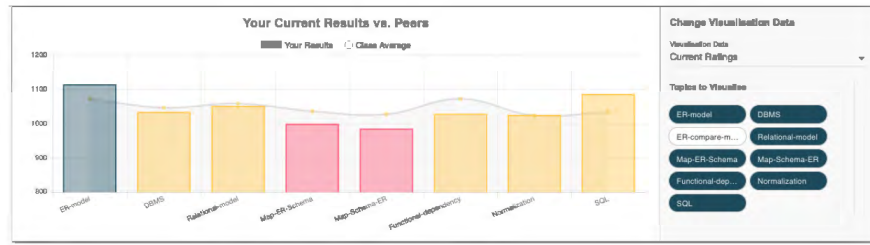


Figure 2: Overview of the OLM visualising the current knowledge state of a student.

that are on concepts where a student has the largest knowledge gap at a difficulty level that reflect the current learning ability of the student. This adaptivity at a concept-level is because of the availability of the additional parameters learned by M-Elo, which is not possible to achieve using the standard Elo rating.

Insights from student feedback. To capture students' perspectives about the model, a survey was conducted at the end of the semester using the following three statements: (1) Motivation: the visualisation used by RiPPLE for showing my knowledge state increases my motivation to study or further use the system, (2) Rationality: having the ability to visually see my current knowledge state, helps me to understand the rationale behind suggestions made by the system, (3) Trust: having the ability to visually see my knowledge state, increases my trust in recommended questions. Responses were captured using the Likert scale, where 1 represents "strongly disagree" and 5 represents "strong agree". The survey also had an open-ended question asking students for feedback on the system. Overall, 55 students who had enrolled in the course and used the system voluntarily participated in the survey. Figure 3 represents the results of the survey. The majority of students (63.6%) agreed or strongly

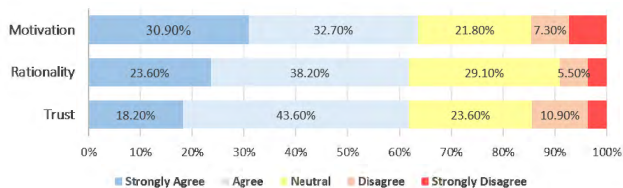


Figure 3: Student survey results

agreed that visualisation of their learner model in RiPPLE increased their motivation to further use the system. Furthermore, the majority of the students (61.8%) agreed that having the ability to visually see their current knowledge state helped them to understand the rationale behind recommendations made by the system and that it increased their trust in recommendations. In their written comments one student mentioned that "I have used a similar platform in the past, however, the visualisation of my knowledge state in this platform is a great improvement on those." Interestingly, two students mentioned that they lost motivation in answering easy questions as the potentially large loss of rating in answering the question incorrectly outweighed the small rating gain received in answering the question correctly. Upon closer examination, we noticed that it seemed

challenging to maintain a balance between the students' proficiency level and the learning items' difficulty level in our pilot. Throughout the semester, the average difficulty level of the learning items was falling and the average rating of the students' proficiency was rising. This can be explained by the student-centred design of the system that provided students full access to the internet, textbooks and colleagues as well as an infinite amount of time for answering a question. This may suggest that the zero-sum game principles where the net change in ratings after a student has answered a question is zero might not be ideal for adaptive learning systems. We believe that this may not be an issue for adaptive testing, where the exam-like setting of the system might balance between the students' ratings and the learning items' ratings.

Insights from instructor feedback. To capture the teaching staff's perspectives, we held informal discussion sessions with members of the teaching staff. They appreciated the additional insight on course level and individual level competencies and gaps that was provided by M-Elo; similar benefits have also been reported by [19]. For example, the learning model presented in Figure 2 indicates that at a class level, students have performed best on "ER-models" and "SQL" and have performed worst on "Map-ER-Schema" and "Map-Schema-ER" in the course presented in this pilot. A shortcoming that the teaching team has noticed was that students tended to get discouraged from using the system once they had used it for a while as they were not able to make significant changes to their knowledge state despite answering questions correctly. Upon closer examination, we realised that this is due to the use of the uncertainty function, as described in section 3 and Formula 4, which reduces the sensitivity of the estimations as the number of attempts is increased. This functionality seems to be better suited for adaptive testing rather than adaptive learning systems because of the following reason: Given that a student would often take an adaptive test in one sitting with a short timeline and receive no feedback on their work, it is common for adaptive testing systems to assume no learning has occurred during the exam, and as such use of an uncertainty function can help with stabilising the ratings; however, a student would often interact with an adaptive learning environment over a period of time and competencies might improve via receiving rich feedback on their learning or decline as a result of forgetting. As a consequence, adaptive systems commonly expect the knowledge state of the student to change significantly as they interact with the system. This means that reducing the sensitivity of the estimations over time may restrict the model from unwaveringly evolving to accurately represent the current knowledge state.

5. CONCLUSIONS AND FUTURE DIRECTIONS

The aim of this paper was to introduce a new multivariate Elo-based learner model called multi-concept multivariate Elo rating system (M-Elo) where learning items can be tagged with one or more concepts and investigate its benefits and shortcomings in the context of adaptive learning. The results from experiments using a suite of synthetic data sets demonstrate that the behaviour of M-Elo is consistent with expectations and outperforms the standard Elo-based model (Elo) in parameter settings that better reflect real-world environments. The results from experiments on multiple benchmarking public data sets indicate that M-Elo has slightly superior predictive performance compared to Elo. Conducting a case study suggested that using M-Elo in adaptive learning systems has additional advantages beyond using Elo. The first advantage lies in the ability of the model in estimating concept-level competencies which can be used for guiding adaptivity. The second advantage lies in the ability of M-Elo to be opened based on the principles of OLMs. Making M-Elo accessible to students increases their motivation to use the platform and increases their trust in the recommendations provided by the platform. It also provides additional insight for instructors on individual student-level or class-level gaps and competencies that can be used to improve item and course design. The conducted case study gave additional insights into the adverse effects of the zero-sum game design of the Elo-based models as well as using an uncertainty functions in the context of adaptive learning. Future work aims to investigate possible extensions of Elo-based models using these considerations and evaluate their fit for adaptive learning systems. In addition, quantitative comparison of M-Elo and traditional models of learner models such as AFM and BKT is required to determine its competitiveness with traditional models in predicting students' performance.

6. REFERENCES

- [1] BODILY, R., KAY, J., ALEVEN, V., JIVET, I., DAVIS, D., XHAKAJ, F., AND VERBERT, K. Open learner models and learning analytics dashboards: a systematic review. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge* (2018), ACM, pp. 41–50.
- [2] BULL, S., AND KAY, J. Open learner models. In *Advances in intelligent tutoring systems*. Springer, 2010, pp. 301–322.
- [3] CEN, H., KOEDINGER, K., AND JUNKER, B. Learning factors analysis—a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems* (2006), Springer, pp. 164–175.
- [4] CORBETT, A. T., AND ANDERSON, J. R. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* 4, 4 (1994), 253–278.
- [5] DE AYALA, R. J. *The theory and practice of item response theory*. Guilford Publications, 2013.
- [6] DESMARAIS, M. C., AND BAKER, R. S. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction* 22, 1-2 (2012), 9–38.
- [7] DESMARAIS, M. C., AND PELCZER, I. On the faithfulness of simulated student performance data. In *Educational Data Mining 2010* (2010).
- [8] DiSALVO, B. B. M. B., ET AL. Khan academy gamifies computer science. In *Proc. 45th ACM Techn. Symp. Comput. Sci. Educ.* (2014), pp. 39–44.
- [9] DOEBLER, P., ALAVASH, M., AND GIESSING, C. Adaptive experiments with a multivariate elo-type algorithm. *Behavior research methods* 47, 2 (2015), 384–394.
- [10] DRASGOW, F., AND HULIN, C. L. Item response theory.
- [11] EMBRETSON, S. E., AND REISE, S. P. *Item response theory*. Psychology Press, 2013.
- [12] KHOSRAVI, H. Recommendation in personalised peer-learning environments. *arXiv preprint arXiv:1712.03077* (2017).
- [13] KHOSRAVI, H., COOPER, K., AND KITTO, K. Ripley: Recommendation in peer-learning environments based on knowledge gaps and interests. *JEDM-Journal of Educational Data Mining* 9, 1 (2017), 42–67.
- [14] KLINKENBERG, S., STRAATEMEIER, M., AND VAN DER MAAS, H. L. Computer adaptive practice of maths ability using a new item response model for on the fly ability and difficulty estimation. *Computers Education* 57, 2 (2011), 1813–1824.
- [15] KOEDINGER, K. R., BAKER, R. S., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., AND STAMPER, J. A data repository for the edm community: The pslc datashop. *Handbook of educational data mining* 43 (2010), 43–56.
- [16] KOEDINGER, K. R., CORBETT, A. T., AND PERFETTI, C. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science* 36, 5 (2012), 757–798.
- [17] PARAMYTHIS, A., AND LOIDL-REISINGER, S. Adaptive learning environments and e-learning standards. In *Second european conference on e-learning* (2003), vol. 1, pp. 369–379.
- [18] PAVLIK JR, P. I., CEN, H., AND KOEDINGER, K. R. Performance factors analysis—a new alternative to knowledge tracing. *Online Submission* (2009).
- [19] PELÁNEK, R. Applications of the elo rating system in adaptive educational systems. *Computers & Education* 98 (2016), 169–179.
- [20] PELÁNEK, R., PAPOUŠEK, J., ŘIHÁK, J., STANISLAV, V., AND NIŽNAN, J. Elo-based learner modeling for the adaptive practice of facts. *User Modeling and User-Adapted Interaction* 27, 1 (2017), 89–118.
- [21] RITTER, S., ANDERSON, J. R., KOEDINGER, K. R., AND CORBETT, A. Cognitive tutor: Applied research in mathematics education. *Psychonomic bulletin & review* 14, 2 (2007), 249–255.
- [22] STAMPER, J., NICULESCU-MIZILM, A., RITTER, S., GORDON, G., AND KOEDINGER, K. Data set from kdd cup 2010 educational data mining challenge, 2010.
- [23] WAUTERS, K., DESMET, P., AND VAN NOORTGATE, W. Monitoring learners' proficiency: weight adaptation in the elo rating system. In *Educational Data Mining 2011* (2010).

How Should Online Teachers of English as a Foreign Language (EFL) Write Feedback to Students?

Cecilia Aguerrebere
Fundación Ceibal, Uruguay
caguerrebere@ceibal.edu.uy

Sofía García
Fundación Ceibal, Uruguay
sgarcia@ceibal.edu.uy

Monica Bulger
Future of Privacy Forum
mbulger@fpf.org

Gabriela Kaplan
Plan Ceibal, Uruguay
gkaplan@ceibal.edu.uy

Cristóbal Cobo
Fundación Ceibal, Uruguay
ccobo@ceibal.edu.uy

Jacob Whitehill
Worcester Polytech. Inst., USA
jrwhitehill@wpi.edu

ABSTRACT

We analyze teachers' written feedback to students in an online learning environment, specifically a setting in which high school students in Uruguay are learning English as a foreign language. How complex should teachers' feedback be? Should it be adapted to each student's English proficiency level? How does teacher feedback affect the probability of engaging the student in a conversation? To explore these questions, we conducted both parametric (multilevel modeling) and non-parametric (bootstrapping) analyses of 27,627 messages exchanged between 35 teachers and 1074 students in 2017 and 2018. Our results suggest: (1) Teachers should adapt their feedback complexity to their students' English proficiency level. Students who receive feedback that is too complex or too basic for their level post 13-15% fewer comments than those who receive adapted feedback. (2) Feedback that includes a question is associated with higher odds-ratio (17.5-19) of engaging the student in conversation. (3) For students with low English proficiency, slow turnaround (feedback after 1 week) reduces this odds ratio by 0.7. These results have potential implications for online platforms offering foreign language learning services, in which it is crucial to give the best possible learning experience while judiciously allocating teachers' time.

1. INTRODUCTION

For decades, teacher feedback has been shown to be one of the greatest drivers of student learning [9]. The research focus has shifted from assessing whether feedback is effective to identifying the most powerful strategies [20]. Because of the complex nature of the feedback process, the answer to this question remains deeply tied to the particular context in which it happens. One particular learning domain that is growing fast in terms of number of learners and learning platforms (e.g., Duolingo, Babbel, Learning English at Coursera) is online learning of English as a foreign language (EFL). Despite its increasing prominence, teacher feedback

in the online EFL context has received limited research attention [20, 5].

In this paper we seek to contribute to the understanding of how teacher feedback influences students' behavior in the online EFL context. In particular, we focus on an EFL program in which students learn English with the help of a remote teacher (RT), who is a native English speaker, with whom students communicate online using discussion forums. Within this context, we seek to build an understanding of how the feedback the RTs give to their students affects their posting behavior: (1) How complex should the RT feedback be? (2) Should it be somehow adapted to their student's English proficiency level? (3) How does RT feedback affect the probability of engaging the student in a conversation? This research has potential implications for the countless online platforms offering foreign language learning services, aiming to enhance students' learning experience.

Learning environment: This study is conducted in the context of a program for EFL learning created for secondary school students who attend the public school system in Uruguay. Uruguayan secondary school students (native Spanish speakers) often struggle with English, having very disparate proficiency levels when they enter high school. This program, known as Tutorials for Differentiated Learning (TDL), was conceived to help tackle this problem by providing the students with the option to learn and practice English at their own pace. For this purpose, a set of resources and exercises for EFL learning are made available online through an LMS system. The students are encouraged by their classroom English teachers (CT) to explore the material and complete the exercises, but participation in the program is not mandatory. Completing an exercise consists of reading the material and posting a comment in English in a discussion forum. Exercises are organized in topics (e.g., music, sports, fashion, national parks, travel, etc) and there is one discussion forum per exercise. A RT, assigned to each classroom, reviews the students' posts and gives them individualized feedback.

RT-student interactions: The student always starts the thread by posting a comment about a given topic in the LMS discussion forum. The RT replies giving the students personalized feedback on what they wrote. Then, the conversation may or may not continue depending on whether the student posts a new comment on the given thread. If the

Cecilia Aguerrebere, Monica Bulger, Cristóbal Cobo, Sofía García, Gabriela Kaplan and Jacob Whitehill "How Should Online Teachers of English as a Foreign Language (EFL) Write Feedback to Students?" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 234 - 239

student doesn't, that conversation ends there, and the student may start new threads when doing new exercises. Here is an example of an interaction where the student engaged in the conversation:

Student: *I do not have favorite music I like to listen to everything a little.*

RT: *That's great Alicia. What's your favourite song right now?*

Student: *At this moment I've heard a song from Michael Jackson that I loved its name is Thriller.*

RT: *Ok Alicia, thank you for sharing that :)*

and another example where she didn't:

Student: *I see six oceans: Atlantic, Indian, Pacific, Atlantic, Arctic and Southern ocean.*

RT: *Very well Andrea.*

Learning English: In TDL, the RTs' feedback is intended less as a way of correcting students' mistakes and more as a way to encourage students to participate. Participation in the discussion forums is expected to be conducive to better learning since doing the exercises requires *reading* the material in English as well as *writing* the response in English. Therefore, two measures of interest are: the total comments the student posts and whether the student engages in a given conversation with the RT.

2. PREVIOUS WORK

Even though there is no unified definition of feedback, the seminal work by Hattie and Timperley [9] conceptualizes feedback as *information provided by an agent regarding aspects of a student's performance or understanding*. It can be provided effectively, but it is dependent on several factors such as the task, the learning context and the learners [9, 20]. It may improve learning outcomes when it has a direct use (e.g. correct the task), or it may increase motivation when only expressing praise for the student [20].

In the online language learning context, feedback has been reported as a fundamental aspect in skills development [11]. Teacher feedback in online language learning environments can also inform development of data-driven personalized feedback. Emerging data-driven learning systems adapt feedback to individual student needs, and have been shown to improve learning outcomes [17]. Furthermore, data mining has been used to understand the effects that polarity (positive vs. negative comments) and timing can have in different student's learning aspects [12, 16].

Research on feedback for EFL learning in computer-mediated (CM) environments has widely focused on *peer* feedback, often on EFL writing [18]. Jiang and Ribeiro [10] present a systematic literature review on the effect of CM peer written feedback on adult EFL writing. They confirmed the findings from previous research acknowledging the positive impact of CM peer feedback in this context.

As in many other subjects in educational data mining, most research on feedback has focused on higher education settings [20], leaving the fundamental need to build understanding of the primary and secondary education contexts unattended. We find previous work on secondary and primary education contexts on particular topics such as teachers' feedback strategies [3], student-generated feedback [8] among other more general examples [15].

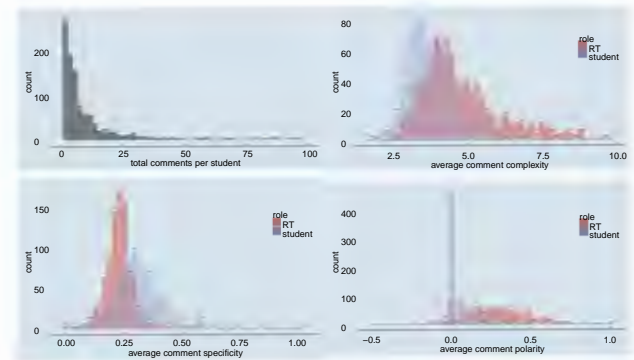


Figure 1: Histogram of the total comments posted by each student (top left). Histograms of the average complexity (top right), specificity (bottom left) and polarity (bottom right) of the RTs' feedback (red) and of the students' posts (violet).

Our work complements and enriches the previous work in several aspects: (1) it studies asynchronous teacher feedback in an online EFL environment, which has been seldom studied [19], (2) it considers a secondary education setting, also fundamental and rarely analyzed [20], (3) it follows a quantitative analysis exploiting a large scale dataset (in terms of number of students, teachers, classrooms, and school diversity) as opposed to most case studies which often include relatively few students or classrooms [19].

3. DATASET DESCRIPTION

The dataset under consideration was originally collected by Aguerrebere et al. [2] and includes all the comments (i.e., content, posting date, user id) as well as administrative information (for each student, who are her CT and RT) for the 1st secondary school classrooms (12-year-olds) that participated in the TDL program during school year 2017. In this work the dataset is extended to also include school year 2018. This includes a total of 27,627 comments exchanged between 1074 students, organized in 83 classrooms (in 49 public high schools located in 18 different states in the country), and 35 RTs. The dataset has a nested structure: students are organized into classrooms. Each classroom has a dedicated classroom teacher; in contrast, each remote teacher may serve multiple classrooms. Figure 1 shows the histogram of the total comments posted by each student during their corresponding school year. The dataset has been de-identified to preserve each participant's privacy and handled according to Uruguayan privacy protection legislation. After talking with the TDL stakeholders and the program leaders, a set of features characterizing each comment was defined: *complexity*, *specificity*, *polarity* and *response delay*. Each feature represents a different aspect of how elaborate a comment is (*complexity*, *specificity*), its tone (*polarity*) and how long the student had to wait to receive feedback (*response delay*).

Complexity (c) measures how elaborated a comment is, by adding its characters per word, words per sentence and total sentences: $c = \frac{1}{4} \frac{\#char}{\#words} + \frac{1}{5} \frac{\#words}{\#sent} + \#sent$ (weights are included to give similar relevance to all terms, 4 and 5 are the median characters per word and words per sentence respectively). Examples of low, medium and high complexity com-

ments are: ($c = 2.4$) “*Well done!*”, ($c = 3.7$) “*My favourite national park is Yellowstone.*”, ($c = 8.9$) “*Hi Alberto! This is an accurate description of the different continents, but can you try again? The activity is asking about different volcanic landforms! Can you please look at the encyclopedia and read the part about volcanic landforms to find the names of the three types of volcanic landforms? Here’s the link: [link]*”.

Specificity (s) measures how specific, on average, the words are in the comment. It combines how deep each word w_i appears in the WordNet [13] structure and how frequent the word is in the dataset: $s = \frac{1}{W} \sum_{i=1}^W \frac{\text{depth}(w_i)}{Z} + \frac{1}{\text{freq}(w_i)}$, where W is the total words in the comment and Z a normalizing factor equal to the maximum average comment complexity [6]. Examples of comments with low and high specificity: ($s = 0.1$) “*Very good! Do you have any cats?*” and ($s = 1.2$) “*The skeleton of brontosaurus.*”.

Polarity (p) measures the tone of the comment (positive, negative) as the average of an index (-1 (negative) to +1 (positive))¹ assigned to each sentence based on the adjectives it contains (e.g., great, nice, awful). Examples of positive and negative comments: ($p = 1.0$) “*Great Carla! Awesome spelling!*” and ($p = -0.6$) “*I would not like because they are dangerous.*”.

Response delay (τ) is the timelapse between the student’s post and the RT’s response in days. Median τ is 3 days.

Figure 1 shows the histograms of the the average *complexity*, *specificity* and *polarity* of the RTs’ feedback and the students’ posts. The average specificity tends to be larger for students than for RTs; this is likely because students’ comments are responses to exercises that elicit very specific words such as “*skeleton of brontosaurus.*”, whereas the RTs’ comments often contain basic words, e.g., “*Great work!*”.

4. DATA ANALYSIS

We examine the effects of various feedback characteristics of RTs’ feedback on students’ posting behavior. We use two complementary approaches [14]: (1) multilevel linear regression that models the nested nature of the data; and (2) non-parametric bootstrap analysis. The latter is more complicated (e.g., requires a bin width parameter) but can model non-linear relationships and makes fewer assumptions (e.g., normality of residuals) than many parametric models.

4.1 How complex should the RT feedback be?

In this section we are interested in the question: Does RT feedback complexity (low vs. high) affect the total number of comments posted by the student? Note that we must consider the potential confound of the student’s English proficiency level, as the effect of RT feedback complexity may vary for more or less proficient students.

4.1.1 Non-parametric approach

To answer this question we first follow a non-parametric approach, using bootstrap to test the null hypothesis:

$$H_0: E[T|S_c] = E[T|S_c, R_c], \quad (1)$$

¹The *sentiment* function of the PATTERN.EN Python module was used: www.clips.uantwerpen.be/pages/pattern-en.

versus $E[T|S_c] \neq E[T|S_c, R_c]$, where T is the total comments posted by the student, S_c is the student’s English proficiency level (low/high) and R_c is the complexity level of the feedback the student received from his RT (low/high). Hypothesis (1) tests whether the total comments posted by the student depend on the complexity level of the feedback she received from her RT, after conditioning on her English proficiency level. If the null hypothesis is rejected, then the complexity level of the feedback that the students receive affects their average engagement with the program, measured by the total comments they post and conditioned on their English proficiency level. It is important to note that we must reject the null hypothesis if $E[T|S_c]$ differs statistically significantly from $E[T|S_c, R_c]$ for *any* values of S_c and R_c . For this reason, in this section we examine the potential impact of R_c on T for each possible combination of (S_c, R_c) .

To estimate the student’s English proficiency level we use the average complexity of the comments posted by the student. Students with average complexity below (above) the median are classified as having low (high) proficiency respectively. Similarly, the complexity level of the RT’s feedback is low (high) when it is below (above) the median.

Bootstrapping for equality of means: How can we test whether $P(T|S_c, R_c = \text{low})$ and $P(T|S_c, R_c = \text{high})$ have the same mean? If the distribution $P(T|S_c, R_c)$ were Gaussian for all values of S_c and R_c , then we could just use a t-test to compute the p -value (or a nested ANOVA to take into account the nested structure of the data). However, in our case the data are not Gaussian (see [1]). Fortunately, the bootstrap procedure proposed by Efron & Tibshirani [7] provides a rigorous methodology. By sampling *with replacement* from our original dataset, we can *simulate* multiple data samples. We subselect the data for which $R_c = \text{low}$ and the data for which $R_c = \text{high}$ and then resample each of them to generate multiple bootstrap samples. To enforce the null hypothesis (i.e., equal means), we *set* the means of the two samples to be equal to the mean of the combined sample. We then compute the normalized difference in means between the two subsets in the bootstrapped sample. Over all B bootstrap iterations ($B = 10000$), we finally compute the fraction in which the normalized difference in means is at least as large as the observed statistic. See [1] for a detailed description of the algorithms.

Results: For high level students, more *basic* feedback is associated with more posting. Students who received high level feedback posted on average 22% fewer comments than those who received low level feedback (9.3 versus 12 total comments, $p = 0.006$). Examples:

Student A: *My favorite food is hamburger.*

RT (low level): *Nice Lucia! What do you eat in your hamburger?*

Student B: *They are in Spain in Barcelona.*

RT (high level): *Hello Pablo! Yes they are in Spain. Very good. Here is a link in case you would like to know a bit more about Spain and their culture. In Uruguay there are a lot of people who have Spanish origins. It is very evident in the food :) I have never been to Spain. Would you like to go to Spain?*

A possible explanation for this behavior is that even the *high-level* students have weak English proficiency and might feel overwhelmed by feedback that is too complex. No statis-

tically significant differences are observed for low level students (8.5 versus 7.6 total comments, $p = 0.14$).

4.1.2 Controlling for the RT

Another possible confound is the effect of the RT, as more motivated RTs may give better feedback that leads to more posts by their students. To take this into account, null Hypothesis (1) is reformulated as:

$$H_0 : E[T|S_c, RT] = E[T|S_c, R_c, RT], \quad (2)$$

versus $E[T|S_c, RT] \neq E[T|S_c, R_c, RT]$, where RT is the student's remote teacher. If Hypothesis (2) is rejected, the complexity level of the feedback *a given RT* gives to his students has an effect on the total comments posted by them. A variation of the bootstrapping algorithm is used to test Hypothesis (2) where, instead of defining the *[low, high]* RT feedback complexity levels globally from all samples, independent thresholds are defined for each RT.

Results: The result previously obtained remains valid even when conditioning on the RT (i.e., resampling within each RT), meaning that different feedback levels given by the RT to his students are associated with different total posts (8.9 versus 12.5 total comments, $p = 0.01$). No statistically significant differences are observed for low level students.

Even if controlling for the RT makes the result more *solid* from a statistical point of view, interpreting the results becomes more difficult, as the meaning of *low* and *high* complexity feedback changes from RT to RT. Because a fundamental goal is to translate these results into useful information for the teachers, this approach has the disadvantage that an *absolute* complexity level reference cannot be given to them as reference of what *low* and *high* means, and how to position the feedback they give with respect to that.

4.1.3 Parametric Approach

A parametric approach is conducted to complement the results obtained by the non-parametric analysis, allowing for the inclusion of additional predictors (possible confounds) and avoiding binning (in RT complexity). Binning is kept to determine student level (low/high), as separate models are computed for each case. In order to take into account the nested structure of the data, a multilevel modeling approach is employed where the CT and RT effects on the student's activity are modeled as nested random effects. Therefore, we model student i 's total posts as a negative binomial random variable, to account for the fact that it is count data with overdispersion, with expected value μ_i given by:

$$\log(\mu_i) = \beta + \gamma_0 c_i + \gamma_1 y_i + \gamma_2 p_i + \gamma_3 s_i + \gamma_4 \tau_i + C + R, \quad (3)$$

(capital letters denote random variables and lower-case denote fixed values). β is the baseline total comments. c_i , p_i , s_i and τ_i are the average complexity, polarity, word specificity and response delay of the feedback comments student i received from his RT, respectively. y_i is the school year. The fixed effects $\gamma_0, \dots, \gamma_4$ represent the effects of the corresponding covariates on the total comments. The nested random effect CT-RT is represented by the random variables C and R , assumed to follow zero-mean Gaussian distributions with standard deviations σ_C and σ_R . All the parametric models were fit using the R *lme4* package [4].

	stud. level	β	$\hat{\gamma}_0$	$\hat{\gamma}_1$	$\hat{\gamma}_2$	$\hat{\gamma}_3$	$\hat{\gamma}_4$
Model 3	low	1.12 **	0.03	0.30 .	-0.03	0.76	0.01
	high	2.50 ***	-0.11 *	0.26	-0.02	-0.02	0.04
Model 4	low	1.63 ***	-0.14 **	0.25	-0.32	0.67	-0.01
	high	2.24 ***	-0.16 **	0.22	-0.06	0.12	-0.002

Table 1: Effects of RT feedback on students' total comments for Models 3 and 4, in log scale. Signif. codes: 0 (*) 0.001 (**) 0.01 (*) 0.05 (.) 0.1**

Results: Table 1 (Model 3) shows the computed effects for all the covariates. The parametric analysis, which includes other possible confounds, confirms the same tendency observed with the non-parametric approach: a negative statistically significant effect of the RT feedback complexity level is observed for high level students ($\exp(-0.11) = 0.9$, i.e., 10% less total posts per unit increase in RT average complexity) and no effect is observed for low level students. To compare this result to the one obtained by the non-parametric approach, we compute the equivalent per unit decrease in the non-parametric case (computed as the total decrease divided by the difference between the average RT complexity in the two compared levels, 3.8 and 5.9) which equals 10%.

4.2 Should RTs feedback complexity be close to that of their students?

Rather than the *absolute* complexity, we can also consider the *relative* complexity of the RTs' feedback compared to the complexity of students' comments. Put another way: should the feedback complexity be somehow *adapted* to the student? To answer this question we propose to model the total comments posted by the student as a function of the *distance* between the average complexity of the student's comments and the average complexity of the feedback the student received from his RT.

4.2.1 Parametric approach

We model each student i 's total posts as a negative binomial random variable with expected value μ_i given by:

$$\log(\mu_i) = \beta + \gamma_0 |c_i - c_{s_i} - \alpha| + \gamma_1 y_i + \gamma_2 p_i + \gamma_3 s_i + \gamma_4 \tau_i + C + R, \quad (4)$$

The fixed effect γ_0 represents the effect of the absolute value of the difference between the student's (c_{s_i}) and the RT's (c_i) average comments complexity, where α is introduced as an offset to account for the fact that the feedback may need to be close to that of the student but not necessarily equal. See Model 3 for the definition of the rest of the variables.

Setting α : Model 4 is fitted for different values of α and the one corresponding to the largest log-likelihood is selected. For low student levels the maximum log-likelihood is obtained at $\alpha = 0.25$, whereas for high student level it is at $\alpha = -0.79$. Hence, this analysis suggests that even if for both low and high level students feedback complexity should be close to the student level, low level students benefit from feedback slightly more complex than theirs whereas high level students benefit from feedback slightly below theirs. Recall that low level students post very basic comments, whereas those of high level students tend to be more elaborated but remain still simple.

Results: Table 1 (Model 4) shows the computed fixed effects for the different covariates. The distance between the average complexity of the student's comments and the average complexity of the feedback the student received from his RT has a negative stat. sig. effect on the total comments posted by the student. There is a 13% ($p = 0.008$) and 15% ($p = 0.003$) decrease in total comments with one unit distance increase, for the low and high level students respectively. We present in the following a series of examples in order to help the reader gain insight into what small and large student-RT complexity distance mean in practice.

Large distance:

Student A: *I would like to defile.*

RT: *Nice try Marcela, but I do not understand what you mean. There are two models in the above photo, can you tell me which model is from Brazil and which model is from the USA? or - can you tell me, who is your favourite model? My favourite model is Kate Moss.*

Small distance:

Student A: *My favorite sport is football.*

RT: *Very good! what is your favorite football team?*

In the latter example, the interactions seem closer to an online chat for students with basic English skills.

4.2.2 Non-parametric approach

A non-parametric approach is conducted to complement the results obtained by the parametric analysis. For this purpose, bootstrapping is used to test the null hypothesis:

$$H_0 : E[T|S_c] = E[T|S_c, D], \quad (5)$$

versus $E[T|S_c] \neq E[T|S_c, D]$, where D is the distance between the student's and the RT's average comments complexity as defined in Model 4. The bootstrapping algorithm introduced in Section 4.1.1 is used, with a for loop on small and large D (below and above the median distance) instead of RT complexity, and α is set to the values obtained in Section 4.2.1 (see [1] for details).

Results: A negative statistically significant effect is observed for D on the total comments posted by the student, both for low and high level students. Students who received feedback less adapted to their level posted 15% and 34% less comments than those who received more adapted feedback, for low (6.6 versus 7.6 total comments, $p = 0.007$) and high (9.2 versus 12.3 total comments, $p < 0.001$) level students respectively. To compare these results to those obtained by the parametric approach we compute the equivalent per unit decrease (computed as the total decrease divided by the difference between the average D in the two compared levels) which equals 9% both for low and high student level.

4.3 Engaging students in conversation

The program aims at motivating the students to interact with others in English. Therefore, we are interested not only in their total posts but also in the probability of engaging them in a conversation. Following the same rationale as Section 4.1.3, we use a multilevel logistic regression model to explore this question. Let $(Y_j)_{j=1,\dots,N}$ be Bernoulli variables with $P(Y_j = 1|\eta_j) = \exp(\eta_j)/(1 + \exp(\eta_j))$ with:

$$\eta_j = \beta + \gamma_0 c_j + \gamma_1 \log(\tau_j) + \gamma_2 p_j + \gamma_3 s_j + \gamma_4 y_j + \gamma_5 q_j + \gamma_6 l_j + \gamma_7 e_j + S + C + R, \quad (6)$$

$Y_j = 1$ if the student posts a second comment in conversation j and 0 otherwise. β is the baseline. c_j , p_j and s_j are the complexity, polarity and specificity of the RT's response to the first comment posted by the student who initiated conversation j . q_j , e_j and l_j are boolean variables taking value 1 if the RT's response asked the student a question, included an emoticon or shared a link respectively. τ_j is the timelapse between the moment the student started conversation j and the RT replied. y_j is the school year. $\gamma_0, \dots, \gamma_7$ represent the fixed effects of the corresponding covariates. The nested random effect student-CT-RT is represented by the random variables S , C and R , assumed to follow zero-mean Gaussian distributions with standard deviations σ_S , σ_C and σ_R . N is the total number of conversation threads and there may be several threads per student.

Results: Table 2 shows the estimated covariate effects. By far, and maybe not surprisingly, the fact that the RT asks the student a question has the largest stat. sig. positive effect on the probability of getting the student to continue the conversation. When a question is asked, assuming the rest of the covariates remain fixed, the odds ratio for students of the same classroom is $\exp(2.94) = 19.0$ and $\exp(2.86) = 17.5$, for low and high level students respectively.

As more elaborated comments often include questions, the positive effect of complexity suggests that more elaborated comments increase the probability of engaging a student in a conversation. On the contrary, the negative stat. sig. effect of polarity is likely due to the fact that very positive comments such as "Great work!" tend to be quite basic in terms of complexity. For high level students a larger delay is associated with more responses, as after a one week delay the odds ratio is 1.2 (the rest of the covariates and random effects remaining constant). This is likely because for high level students RT response delay is positively correlated with complexity (Spearman 0.1) and negatively correlated with comments polarity (Spearman -0.12): writing more elaborated comments take longer. This is not observed for low level students (Spearman 0.02 for both complexity and polarity), for whom it seems important to reply as soon as possible, as after a one week delay the odds ratio is 0.7. Finally, for both high and low level students, results suggest that using less specific words is associated with higher probability of engagement in the conversation.

5. SUMMARY AND CONCLUSIONS

We conducted an observational analysis of 27,627 comments, exchanged between 1074 high school students and 35 RTs over 2 years, to study the effect that different RT's feedback characteristics have on the students' posting behavior in an online EFL learning environment. The research questions, as well as the features defined for the characterization of the comments, were discussed and validated with the stakeholders and the leaders of the program in order to take advantage of their wide experience on the topic. Both parametric (multilevel modeling) and non-parametric (bootstrapping) analyses were performed, controlling for the effect of possible confounds such as (1) the classroom teacher, (2) the remote teacher and (3) the students' English proficiency level. Our results suggest that:

(1) Teachers should observe the complexity of their students' comments and adapt the complexity of their feedback ac-

student level	β	$\hat{\gamma}_0$	$\hat{\gamma}_1$	$\hat{\gamma}_2$	$\hat{\gamma}_3$	$\hat{\gamma}_4$	$\hat{\gamma}_5$	$\hat{\gamma}_6$	$\hat{\gamma}_7$
low	-7.11 ***	0.08	-0.15 ***	0.36 ***	-2.20 **	0.24	2.94 ***	-0.53	0.11
high	-7.46 ***	0.35 ***	0.10 **	-0.47 *	-2.66 **	0.62	2.83 ***	-0.77	-0.39

Table 2: Effects of RT feedback characteristics on the probability of engaging the student in conversation, in logarithmic scale. Significance codes: 0 (*) 0.001 (**) 0.01 (*) 0.05 (.) 0.1**

cordingly. Students who receive feedback that is too complex or too basic for their level post 13% ($p = 0.008$) and 15% ($p = 0.003$) fewer comments than those who receive adapted feedback, for low and high level students respectively.

(2) According to some RTs who were consulted about the potential causes of the observed behavior, the students may be more motivated when the language of the RT is accessible to them because they understand it, they learn from it and are challenged by it, without this turning into frustration.

(3) The best way to engage the students in a conversation is to pose a question (this increases the odds by 19 and 17.5 for low and high level students respectively). The comments should be complex enough to include a question (i.e., “Great work!” won’t be enough) yet remain simple in terms words specificity. Also, for low level students, it is important to respond as quickly as possible (after a one week delay the odds ratio is 0.7).

Even if no causal inferences can be made, this study generated enlightening insights which have potential implications for the countless online platforms offering foreign language learning services, in which it is crucial to give the best possible learning experience while judiciously allocating resources (e.g. teachers’ time).

Acknowledgments: We thank the TDL team and RTs for their participation and helpful feedback. J. Whitehill was supported by the National Science Foundation under Grant No. #1822830.

6. REFERENCES

- [1] AGUERREBERE, C., BULGER, M., COBO, C., GARCÍA, S., KAPLAN, G., AND WHITEHILL, J. How should online teachers of english as a foreign language write feedback to students? Available at https://users.wpi.edu/~jrwhitehill/AguerreberEtAl_EDM2019_full.pdf (2019).
- [2] AGUERREBERE, C., CABEZA, S. G., KAPLAN, G., MARCONI, C., COBO, C., AND BULGER, M. Exploring feedback interactions in online learning environments for secondary education. In *Proc. of Latin Amer. Workshop on Learn. Analytics* (2018), pp. 128–137.
- [3] BAADTE, C., AND KURENBACH, F. The effects of expectancy-incongruent feedback and self-affirmation on task performance of secondary school students. *Eur. J. Psychol. Educ.* 32, 1 (2017), 113–131.
- [4] BATES, D., MÄCHLER, M., BOLKER, B., AND WALKER, S. Fitting linear mixed-effects models using lme4. *J Stat Softw* 67, 1 (2015), 1–48.
- [5] CONRAD, S. S., AND DABBAGH, N. Examining the factors that influence how instructors provide feedback in online learning environments. *Int. J. of Online Pedagogy and Course Design* 5, 4 (2015), 47–66.
- [6] DESHPANDE, S., PALSHIKAR, G. K., AND ATHIAPPAN, G. An unsupervised approach to sentence classification. In *COMAD* (2010), p. 88.
- [7] EFRON, B., AND TIBSHIRANI, R. J. *An introduction to the bootstrap*. CRC press, 1994.
- [8] HARRIS, L. R., BROWN, G. T., AND HARNETT, J. A. Analysis of New Zealand primary and secondary student peer-and self-assessment comments: Applying Hattie and Timperley’s feedback model. *Assessment in Education: Principles, Policy & Practice* (2015).
- [9] HATTIE, J., AND TIMPERLEY, H. The power of feedback. *Rev. Educ. Res.* 77, 1 (2007), 81–112.
- [10] JIANG, W., AND RIBEIRO, A. Effect of computer-mediated peer written feedback on ESL/EFL writing: A systematic literature review. *Elec. Int. J of Educ., Arts, and Science* 3, 6 (2017).
- [11] KAHRAMAN, A., AND YALVAC, F. EFL turkish university students’ preferences about teacher feedback and its importance. *Procedia-Social and Behavioral Sciences* 199 (2015), 73–80.
- [12] LANG, C., HEFFERNAN, N., OSTROW, K., AND WANG, Y. The impact of incorporating student confidence items into an intelligent tutor: A randomized controlled trial. *Int. Educ. Data Mininig Soc.* (2015).
- [13] MILLER, G. A. Wordnet: a lexical database for english. *Comm. of the ACM* 38, 11 (1995), 39–41.
- [14] MIYAMOTO, Y., COLEMAN, C., WILLIAMS, J., WHITEHILL, J., NESTERKO, S., AND REICH, J. Beyond time-on-task: The relationship between spaced study and certification in MOOCs. *Available at SSRN 2547799* (2015).
- [15] OINAS, S., VAINIKAINEN, M.-P., AND HOTULAINEN, R. Technology-enhanced feedback for pupils and parents in finnish basic education. *Computers & Education* 108 (2017), 59–70.
- [16] OLSEN, J. K., ALEVEN, V., AND RUMMEL, N. Predicting student performance in a collaborative learning environment. *Int. Educ. Data Mining Soc.* (2015).
- [17] ROMERO, C., AND VENTURA, S. Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 3, 1 (2013), 12–27.
- [18] SAEED, M. A., GHAZALI, K., AND ALJABERI, M. A. A review of previous studies on ESL/EFL learners’ interactional feedback exchanges in face-to-face and computer-assisted peer review of writing. *Int. J. of Educ. Tech. in Higher Education* 15, 1 (2018), 6.
- [19] SHANG, H.-F. An exploration of asynchronous and synchronous feedback modes in EFL writing. *J. of Computing in Higher Education* 29, 3 (2017), 496–513.
- [20] VAN DER KLEIJ, F. M., FESKENS, R. C., AND EGGEN, T. J. Effects of feedback in a computer-based learning environment on students learning outcomes: A meta-analysis. *Rev. Educ. Res.* 85, 4 (2015).

Concept-Aware Deep Knowledge Tracing and Exercise Recommendation in an Online Learning System

Fangzhe Ai

School of Electronics and
Information Engineering
Beijing Jiaotong University
17125001@bjtu.edu.cn

Yongxiang Zhao

School of Electronics and
Information Engineering
Beijing Jiaotong University
yxzhao@bjtu.edu.cn

Yishuai Chen

School of Electronics and
Information Engineering
Beijing Jiaotong University
yschen@bjtu.edu.cn

Zhenzhu Wang

School of Electronics and
Information Engineering
Beijing Jiaotong University
18120144@bjtu.edu.cn

Yuchun Guo

School of Electronics and
Information Engineering
Beijing Jiaotong University
ycguo@bjtu.edu.cn

Guowei Fu

TAL Education Group Inc.
Beijing, China, 100080
fuguowei@100tal.com

ABSTRACT

Personalized education systems recommend learning contents to students based on their capacity to accelerate their learning. This paper proposes a personalized exercise recommendation system for online self-directed learning. We first improve the performance of knowledge tracing models. Existing deep knowledge tracing models, such as Dynamic Key-Value Memory Network (DKVMN), ignore exercises' concept tags, which are usually available in tutoring systems. We modify DKVMN to design its memory structure based on the course's concept list, and explicitly consider the exercise-concept mapping relationship during students' knowledge tracing. We evaluated the model on the 5th grade students' math exercising dataset in TAL, one of the biggest education groups in China, and found that our model has higher performance than existing models. We also enhance the DKVMN model to support more input features and obtain higher performance. Second, we use the model to build a student simulator, and use it to train an exercise recommendation policy with deep reinforcement learning. Experimental results show that our policy achieves better performance than existing heuristic policy in terms of maximizing the students' knowledge level. To the best of our knowledge, this is the first time that deep reinforcement learning has been applied to personalized mathematic exercise recommendation.

1. INTRODUCTION

Online self-directed learning systems, such as Massive Open Online Courses (MOOCs), are prevailing. These systems, however, assign same exercises to all students, which is inefficient. For comparison, personalized exercises recommendation can improve the efficiency of students' learning. In

this paper, we propose a personalized exercise recommendation system for an online self-directed learning service. The system consists of two parts:

- A student knowledge tracing model, which traces a student's knowledge state and predicts whether or not she can finish the exercise correctly.
- A personalized exercise recommendation policy which recommends appropriate exercises to students to accelerate her learning process.

Existing deep knowledge tracing models [9, 13] ignore exercises' knowledge concept properties, which are usually available in tutoring systems. For comparison, in this paper, we propose a concept-aware deep knowledge tracing model. The model is inspired by Dynamic Key-Value Memory Network (DKVMN) model [13]. DKVMN model has a static matrix called *key* which stores the latent knowledge concepts and a dynamic matrix called *value* which stores a student's concept mastery levels. The model computes the correlation between an exercise and the latent concepts in the *key*, and then uses it to read the student's concept mastery levels in the *value*, and predict whether the student will finish the exercise correctly. We improve the DKVMN model as follows: 1) we design its memory structure based on the course's concept list and explicitly consider the exercise-concept mapping relationship during students' knowledge tracing. 2) We enhance it to support more input features, including exercise difficulty, stages, and student practice time. We evaluated the model on the 5th grade students' math exercising dataset in TAL, and found that our model has higher performance than existing deep knowledge tracing models.

In terms of personalized exercise recommendation policy, most of existing algorithms are heuristic, e.g., exercises which are too easy or too hard for a student should be avoided. These algorithms may be not optimal, as they only consider the short-term reward. In this paper, we build a student simulator with our concept-aware deep knowledge tracing model, and then use it to train a flexible and scalable personalized exercise recommendation policy with deep re-

Fangzhe Ai, Yishuai Chen, Yuchun Guo, Yongxiang Zhao, Zhenzhu Wang, Guowei Fu and Guangyan Wang "Concept-Aware Deep Knowledge Tracing and Exercise Recommendation in an Online Learning System" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 240 - 245

inforcement learning, which considers long-term reward of recommended items.

In summary, the main contributions of this paper are two folds:

- We propose a new exercise-level deep knowledge tracing model whose structure is built based on the course's concept list, and the exercise-concept mapping relationships are utilized during students' knowledge tracing. The model supports more input features and obtains higher performance compared with existing models.
- We propose an exercises recommendation algorithm which uses model-free reinforcement learning with neural network function approximation to learn an exercise recommendation policy. The policy directly operates on raw observations of a student's exercise history. Experimental results show that our policy achieves better performance than existing heuristic policy in terms of maximizing students' knowledge level.

2. RELATED WORK

Knowledge Tracing: The work in [3] proposed a Bayesian-based knowledge tracing model. It models a student's status of a knowledge concept as a binary variable, and updates the probability of her mastering the concept according to her results of doing exercises through a Hidden Markov Model. This model is at the concept level, and ignores the relationship between different concepts. The work in [9] proposed a deep knowledge tracing (DKT) model with recurrent neural network. It models a student's knowledge states as latent variables, and gets better performance than Bayesian-based model does [6]. The work in [12] proposed to improve DKT by considering exercises' semantic features. The work in [13] tried to model the correlation between different latent concepts. Inspired by DKVMN, this paper proposes a model whose structure is explicitly built based on the course's concept list, and the exercise-concept mapping relationship is utilized in the model.

Exercise Recommendation: The work in [1] proposed that a student is recommended by an exercise, if the probability of her doing the exercise correctly is around 50%. The problem of this algorithm is that the threshold 50% is heuristic and may be not optimal. The work in [2] allows experts to specify a ZPD (Zone of Proximal Development) based on current knowledge state of a student, and then chooses the most profitable exercise by multi-armed bandits algorithm. The algorithm can discover the characteristics of students through exploration but it is inefficient, because every student needs an independent exploration process. The work in [5] leverages a DKT model towards recommendation, and frame the problem space using ZPD explicitly facilitated by the DKT model. The work in [7] first estimates each student's knowledge profile from their previous exercise results using SPARFA framework. Then, it uses these knowledge profiles as contexts and applies contextual bandits algorithm to recommend exercises, for maximizing a student's immediate success, i.e., her performance on the next exercise. The problem of this algorithm is that it only considers the next step and thus its performance may be

not optimal. The work in [10] evaluated a review scheduling algorithm for spaced repetition systems based on deep reinforcement learning. We are inspired by this work and evaluate the performance of deep reinforcement learning in our math self-directed learning system.

3. BACKGROUND

In this section, we introduce our online learning system and dataset.

3.1 Intelligent Practice System (IPS)

IPS is an online self-directed learning system developed by TAL Education Group, Inc. of China. In IPS, each course (e.g., the 5th grade math) has tens of units. Each unit includes 7 stages, i.e., 1) warming-up exercises before class, 2) in-class exercises before lecture, 3) video lecture, 4) in-class exercises after lecture, 5) homework exercises, 6) unit review exercises, 7) multi-units review exercises. In these 7 stages, stages 1, 2, 3, 4, 5 include contents of a single knowledge concept, but stages 6 and 7 include exercises of other knowledge concepts in order to review. As IPS is a self-directed learning system, a student can choose any teaching unit to study. In a unit, she can also exit current stage or the whole unit at any time. The system records the student's learning duration in each stage, the exercises she practices, and her results, i.e., whether or not the answer is correct.

In IPS, each exercise has three knowledge concept tags, which are provided by experts. The knowledge concept tags have a hierarchical tree structure. For instance, for one exercise, its 1st, 2nd, and 3rd level concept tags are "Number Theory", "Prime Number and Composite Number", and "Decomposition of Prime Factor", respectively.

3.2 Data Set and Data Pre-Processing

We use a sample of anonymized student usage interactions from the 5th grade math curriculum in IPS. We choose exercising records whose first-level knowledge concept is "Number Theory", which has 7 second-level knowledge concepts and 15 third-level knowledge concepts. We further choose students whose exercise records include at least 5 exercises. The resulting dataset includes 44,128 exercise records of 7,124 students.

4. KNOWLEDGE TRACING MODEL

We now introduce our knowledge tracing model based on DKVMN model, and highlight our improvement in aspects of memory structure, knowledge concept weight, and read and update process.

4.1 Concept-Aware Memory Structure

We modify DKVMN to design its memory structure based on the course's concept list. Fig. 1 plots the model's structure, which is based on DKVMN model [13]. As shown in Fig. 1, \mathbf{M}_t^k is the concept embedding matrix whose size is $M \times N$, where N is the number of memory locations, and M is the vector size at each location. We set N equal to the number of the course's knowledge concepts. As we have 1 first-level knowledge concept, 7 second-level knowledge concepts and 15 third-level knowledge concepts, we have $N = 23$. Then, in each location, the student's state for the corresponding knowledge concept is saved. Thus, the

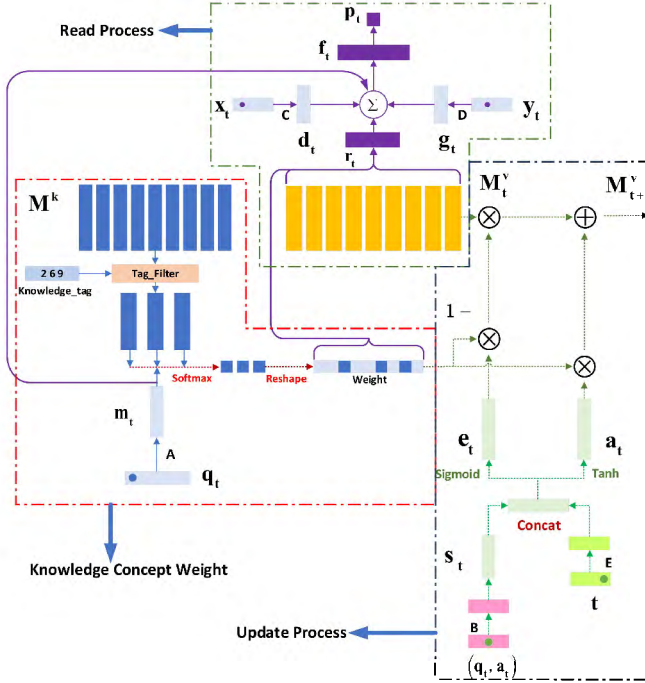


Figure 1: Concept-aware DKVMN model structure.

model's memory architecture is explicitly designed to represent knowledge concepts. For comparison, N is a model parameter in DKVMN representing the number of latent knowledge concepts, e.g., $N = 5$. As our model is inspired by DKVMN, we name it Concept-Aware DKVMN, i.e., *DKVMN-CA*.

4.2 Knowledge Concept Weight

As a student's state of a knowledge concept is saved in the corresponding memory location, when a new exercise arrives, only the exercise's related concepts' memory locations are retrieved and updated. We now present the details of such a procedure. In this section, we calculate the knowledge concept weight (KCW) of the exercise. The weights will be used to calculate the weighted sum of a user's current knowledge concept states to predict her performance on the exercise. It will also be used to update the student's knowledge state after obtaining the answer result of the student on the exercise.

We first obtain the embedding of the arrived exercise. As shown in Fig. 1, when an exercise q_t arrives at time t , it is first transformed into an embedding vector m_t through an exercise embedding matrix A . We then calculate the KCW through Algorithm 1. As shown in Algorithm 1, at line 2, we initialize the weight list R . As each exercise has three knowledge concepts, the length of R is 3. Then, for each knowledge concept k (line 2), we calculate the dot product of the embedding of the exercise (i.e., q_t) and the concept embedding (line 3). We then calculate the KCW by obtaining softmax of R , with $\text{Softmax}(z_i) = e^{z_i} / \sum_{j=1}^N e^{z_j}$ (line 6). Then, we initialize an all-zero vector *Weight* whose length is the number of the knowledge concepts N (line 7). For each knowledge concept k of the exercise (line 8), we set

its weight value in *Weight*.

In summary, DKVMN computes the relationship weights between the exercise and all latent knowledge concepts, but we just compute the relationship weights between the exercise and its knowledge concepts. For the exercise's relationship weights with other concepts, we set them zeros.

Algorithm 1 Knowledge Concept Weight Calculation

Input:
 q_t : embedding of the exercise arrived at time t
 K_t : knowledge concept list of q_t
 M^k : the concept embedding matrix

Output:
Weight: Knowledge concept weight of the exercise arrived at time t

/ Calculate KCW */*

- 1: $R \leftarrow []$
- 2: **for** each $n \in K_t$ **do**
- 3: $corr \leftarrow m_t^T \cdot M^k[n]$
- 4: $R.append(corr)$
- 5: **end for**
- 6: $R_s \leftarrow \text{Softmax}(R)$
- /* Reshape the weight vector to make its length equal to the number of concepts */*
- 7: $Weight \leftarrow [0, \dots, 0]$
- 8: $i \leftarrow 0$
- 9: **for** $i < 3$ **do**
- 10: $Weight[K_t[i]] \leftarrow R_s[i]$
- 11: $i \leftarrow i + 1$
- 12: **end for**
- 13: **return** *Weight*

4.3 Read Process

We then use the obtained KCW to calculate the weighted sum of the user's current knowledge concept states to predict the student's performance on the exercise. Denote KCW by w , we have $r_t = \sum_{i=1}^N w_i M_t^v$, i.e., the knowledge state of concepts related to the exercise q_t .

We further concatenate r_t with the embeddings of the exercise's difficulty and stage feature, i.e., d_t and g_t . The result then passes through a fully connected layer with activation function Tanh to get a summary vector f_t , which contains all information of the student's knowledge state related to q_t and the exercise's features, i.e.,:

$$f_t = \text{Tanh}(W_0^T [r_t, d_t, g_t, m_t])$$

where $\text{Tanh}(z_i) = (e^{z_i} - e^{-z_i}) / (e^{z_i} + e^{-z_i})$.

Finally, f_t passes through a fully connected layer to output the probability that the student would do the exercises q_t correctly. Denote the probability by p , we have

$$p = \text{Sigmoid}(W_1^T f_t)$$

where $\text{Sigmoid}(z_i) = 1 / (1 + e^{-z_i})$.

4.4 Update Process

We then use the KCW to update the student's knowledge state after observing the her answer result. The update process updates the value matrix M_t^v , which represents the student's current state of knowledge concept k . Our model is different from DKVMN model in that we consider the student's exercising duration in the update process. For

comparison, DKVMN ignores this student behavior feature. Specifically, the work in [8] proposed that the a student's duration of solving a problem is related to her master level of latent problem solving skills. Inspired by this work, in our model, after a student finishes an exercise, her answer result (i.e., correct or wrong) \mathbf{a}_t and exercising duration are used to update \mathbf{M}_t^v . Because the exercising duration is a continuous variable, it is firstly discretized according to its distribution and then represented by its embedding \mathbf{t} . We then concatenate \mathbf{t} with the joint embedding \mathbf{s}_t of the answer vector $(\mathbf{q}_t, \mathbf{a}_t)$, to update \mathbf{M}_t^v , as shown in Fig. 1.

The other update process is same as that of DKVMN. It includes erase subprocess and add subprocess. Erase vector is computed as $\mathbf{e} = \text{Sigmoid}(\mathbf{E}^T[\mathbf{s}_t, \mathbf{t}])$, where \mathbf{E} is the erase weights. Add vector is computed as $\mathbf{a} = \text{Tanh}(\mathbf{D}^T[\mathbf{s}_t, \mathbf{t}])$, where \mathbf{D} is the add weights. Then the new memory matrix \mathbf{M}_{t+1}^v is computed by

$$\mathbf{M}_{t+1}^v(i) = \mathbf{M}_t^v(i)[1 - \mathbf{w}(i)\mathbf{e}][1 + \mathbf{w}(i)\mathbf{a}]$$

The parameters of the model are learned by minimizing a standard cross entropy loss between the predicted user answer result p_t and her true result y_t :

$$L = - \sum_t ((y_t \log p_t) + (1 - y_t) \log(1 - p_t))$$

In summary, compared with DKVMN, we design the model structure based on the course's concept list, and then explicitly consider the exercise-concept mapping relationship and other exercise's features during students' knowledge tracing.

5. REINFORCEMENT LEARNING BASED EXERCISES RECOMMENDATION

Based on the DKVMN-CA student knowledge tracing model, we build a student simulator which provides environment for reinforcement learning, and train a personalized exercise recommendation agent with deep reinforcement learning method.

Similar to [10], we model the recommendation process as a Partially Observable Markov Decision Process (POMDP), where the model state is the student's latent knowledge state and the action is the recommendation of an exercise. At time t , the reinforcement learning agent cannot observe the student's latent knowledge state s_t . Instead, it can observe the student's exercise and answer result (i.e., correct or wrong) o_t which is conditioned on the latent knowledge state $p(o_t|s_t)$. Thus, at time t , the agent needs to recommend an exercise a_t based on the student's exercising history before t , which is denoted by h_t . We have $h_t = (o_1, a_1, o_2, a_2, \dots, o_{t-1}, a_{t-1})$. After the student finishes the recommended exercise a_t , her latent knowledge state will turn to s_{t+1} by a transition function $p(s_{t+1}|s_t, a_t)$.

We define the reward r_t of an action a_t as

$$r_t = \frac{1}{K} \sum_{i=1}^K P_{t+1}(q_i), \quad (1)$$

where K is the number of exercises, and $P_{t+1}(q)$ denotes the probability of the student getting exercise q correct after finishing the recommended exercise at state s_{t+1} . It is

predicted by the student simulator. So, we name it as the student's *Predicted Knowledge*.

The purpose of optimization is to maximize the reward R of policy π :

$$R = \mathbb{E}_\pi \left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t) \right],$$

where trajectories $\tau = (s_1, o_1, a_1, s_2, o_2, a_2, \dots)$ are drawn from the trajectory distribution induced by policy $\pi : p(s_1) p(o_1|s_1) \pi(a_1|h_1) p(s_2|s_1, a_1) p(o_2|s_2) \pi(a_2|h_2) \dots$. Thus, as for the action-value function Q^π , the reward of the recommended exercise sequence at t is:

$$Q^\pi(h_t, a_t) = \mathbb{E}_{s_t|h_t} [r_t(s_t, a_t)] + \mathbb{E}_{\tau>t|h_t, a_t} \left[\sum_{i=1}^{\infty} \gamma^i r(s_{t+i}, a_{t+i}) \right]$$

where $\tau > t = (s_{t+1}, o_{t+1}, a_{t+1}, \dots)$ is the future trajectory. The algorithm then recommends the exercise q_t which has the maximal reward, i.e., $q_t = \max_a Q^\pi(h_t, a)$. Similar to [10], we approximately solve the POMDP using Trust Region Policy Optimization (TRPO) algorithm [11], with an off-the-shelf implementation from rllab [4].

6. PERFORMANCE EVALUATION

In this section, we present the performance evaluation results of our system.

6.1 DKVMN-CA Knowledge Tracing Model

We evaluated our model on our IPS dataset. To evaluate it, we conducted 50 experiments. In each experiment, we randomly split the users into two groups: training users and testing users. Their percentages are 70% and 30%. We then trained the model with the training users and evaluated the model on the testing users. Similar to [9], we use area under the curve (AUC) as the performance metric. We report the maximal, mean, and the standard deviation of the testing users' AUCs of all 50 experiments.

6.1.1 Efficiency of Concept-Aware Design

We first report the efficiency of designing the model's architecture based on the course's knowledge concepts. We compare the performance of DKVMN and DKVMN-CA without the help of other input features, including exercise difficulty, stage, and duration. The results are shown in Table. 1. See the rows "DKVMN" and "DKVMN-CA". As shown in Table. 1, our model obtains an AUC of 0.724, which is higher than that of DKVMN model, i.e., AUC = 0.712. Such an improvement is considerable, considering the small improvement DKVMN provides over the DKT baseline (AUC = 0.711). Such a result means that the design of the model's architecture based on the course's knowledge concepts is efficient.

To highlight the necessity of our design, we also evaluate another method which also uses exercises' knowledge concept tags, i.e., represents a knowledge concept by its embedding and then concatenate it with the embedding of the exercise. Its' performance is shown in Table. 1. See the row "DKVMN-KC". As shown in Table 1, its mean AUC is 0.714, meaning a very small improvement over the DKVMN (AUC = 0.712). Thus, it is necessary to design the model's

Table 1: AUC of Models with Different Features

Model	Mean AUC	Max AUC	Variance
LSTM	0.711	0.712	1.86e-05
DKVMN	0.712	0.720	2.05e-05
DKVMN-KC	0.714	0.724	1.85e-05
DKVMN-CA	0.724	0.731	2.14e-05
DKVMN-CA + Stage	0.728	0.736	1.48e-05
DKVMN-CA + Duration	0.725	0.737	1.75e-05
DKVMN-CA + Difficulty	0.726	0.736	2.44e-05
DKVMN-CA + Stage, Duration	0.726	0.739	2.43e-05

architecture based on the course’s knowledge concepts to fully utilize their capacity to improve the model.

6.1.2 Efficiency of Other Exercise Features

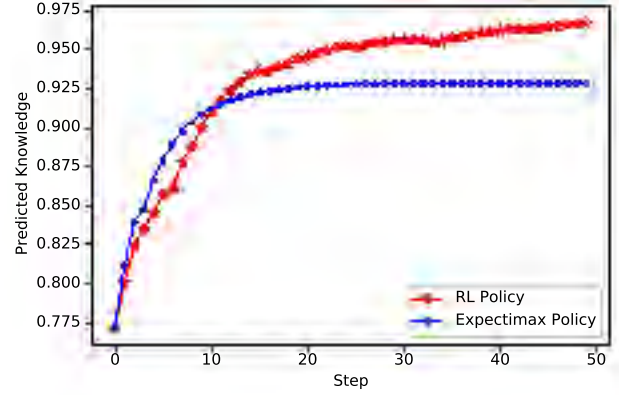
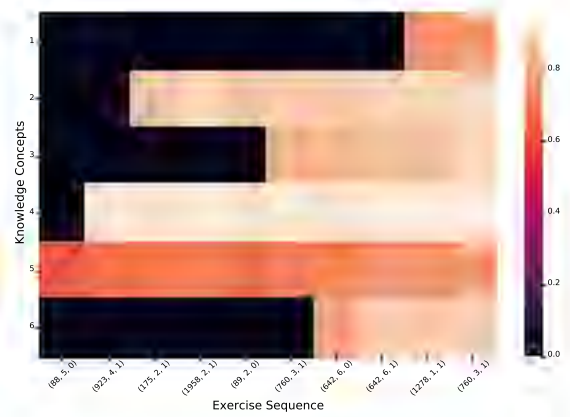
We then evaluate the efficiency of adding other features, including exercise difficulty, stage, and duration. The results are shown in Table 1. See the rows "DKVMN-CA + Difficulty", "DKVMN-CA + Stage", and "DKVMN-CA + Duration". As shown in Table 1, these features can further improve the model’s performance. For example, the mean AUC of "DKVMN-CA + Stage" is 0.728, which is higher than that of DKVMN model (AUC = 0.712).

6.2 Exercises Recommendation

6.2.1 Evaluation of Students’ Knowledge Growth Process

We use the Expectimax algorithm proposed in [9] as the baseline algorithm. In the Expectimax algorithm, the system first calculates a student’s predicted knowledge assuming an exercise is recommended to the user to practice. It then chooses the exercise with the highest predicted knowledge to recommend.

To compare the two algorithms, we first randomly pick 15 students in our dataset. For each student, we conduct two experiments, one experiment for one algorithm. In each experiment, we first initialize the student simulator using the student’s historical practice sequence. Then, we continuously recommend 50 exercises to the student simulator with the recommendation algorithm. During the process, we record the average of the 15 students’ predicted knowledge as Eq.(1) at each step of recommendation. The results are shown in Fig. 2. As shown in Fig. 2, the students served by RL policy has a higher mean predicted knowledge than the students served by the Expectimax policy after 50 exercises. Moreover, after about 10 exercises, the mean predicted knowledge of the students served by the Expectimax policy stops increasing, meaning that the policy cannot find exercises which can help the students to improve the performance any more. For comparison, served by the RL policy which considers long-term reward of action, the students’ mean predicted knowledge keeps increasing, meaning that the RL policy still can find exercises which can help the student to improve performance.

**Figure 2: Performance evaluation results****Figure 3: Knowledge State Variation**

6.2.2 Evaluation of Recommendation Process

We design another experiment to observe the recommendation behavior of the RL policy. We randomly pick a student who has practiced five exercises, and use her exercise sequence to initialize the student simulator. Then, we serve her with five more exercises using the RL recommendation policy. Fig. 3 shows the results. The x label of Fig. 3 shows the 10 exercises’ IDs, concepts, and results. For example, the first record (88, 5, 0) means that the exercise ID is 88, which is related to concept 5, and the student’s answer is wrong. As the 10 exercises are related to 6 concepts, we plot the student’s predicted knowledge of each concept in Fig. 3. For instance, as the student fails in the first exercise 88, which is related to the 5th concept, the student’s knowledge status on the 5th concept is relatively low. The status of the knowledge concepts not covered by the student’s history exercises are indicated in black. We now observe the recommended exercises. We have the following observations:

- As shown in Fig. 3, the first five exercises are related to concepts 2,4,5, and the later five exercises are related to concepts 1,3,6, suggesting the RL algorithm wants to explore the student’s capacity in other concepts.

- After the student succeeds in exercise 760, which is related to the concept 3 (Decomposition of Prime Factor), the algorithm recommends the exercise 642, which is related to concept 6 (Maximum common factor and Least common multiple). As concept 6 is related to concept 3, such a recommendation is reasonable.
- The student, however, fails to finish the exercise 642. Thus, the algorithm recommends exercise 642 again. This time, the student succeeds to finish it, meaning that the model captures the phenomenon during training that a student who failed in exercise 642 may succeed if she retries. Such a result is interesting.
- After the student succeeds in exercise 642, which is related to concept 6 (Maximum common factor and Least common multiple), the model's estimation of the student's capacity on concept 3 (Decomposition of Prime Factor) also slightly increases. As these two concepts are indeed related, such a result is reasonable.
- Then, the algorithm turns to another concept again, i.e., it recommends the exercise 1278, which is related to concept 1. While the student succeeds in the exercises, the estimated student's knowledge status on the concept 1, however, is relatively low, suggesting that the exercise is relative easy.
- At last, the exercise 760 is recommended again, and the student succeeds in it. As a result, the model's estimation of the student's capacity on concept 3 increases, suggesting that reviewing is beneficial for study.

7. CONCLUSION

In this paper, we improve DKVMN by designing its neural network structure based on a course's concept list, and explicitly considering the exercise-concept mapping relationship during students' knowledge tracing. We also enhance the DKVMN model to consider more input features. Experimental results show that our model has higher performance than existing deep knowledge tracing models. We also propose an exercises recommendation algorithm which uses model-free reinforcement learning with neural network function approximation to learn an exercise recommendation policy that directly operates on raw observations of a student's exercise history. Our experimental results demonstrate that our policy achieves better performance than existing heuristic policy in terms of maximizing the students' knowledge level. To the best of our knowledge, this is the first time that deep reinforcement learning has been applied to personalized mathematic exercise recommendation.

8. ACKNOWLEDGEMENT

This work was supported by National Natural Science Foundation of China under Grants 61572071, 61872031, and 61301082.

9. ADDITIONAL AUTHORS

Additional authors: Guangyan Wang (College of Education, Hebei University, email: gywang@163.com).

10. REFERENCES

- [1] I.-A. Chounta, B. M. McLaren, P. L. Albacete, P. W. Jordan, and S. Katz. Modeling the zone of proximal development with a computational approach. *EDM*, 2017:56–57, 2017.
- [2] B. Clement, D. Roy, P.-Y. Oudeyer, and M. Lopes. Multi-armed bandits for intelligent tutoring systems. *arXiv preprint arXiv:1310.3174*, 2013.
- [3] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [4] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.
- [5] W. Jiang, Z. A. Pardos, and Q. Wei. Goal-based course recommendation. pages 36–45, 2018.
- [6] M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing? *arXiv preprint arXiv:1604.02416*, 2016.
- [7] A. S. Lan and R. G. Baraniuk. A contextual bandits framework for personalized learning action selection. In *EDM*, pages 424–429, 2016.
- [8] R. Pelánek and P. Jarušek. Student modeling based on problem solving times. *International Journal of Artificial Intelligence in Education*, 25(4):493–519, 2015.
- [9] C. Piech, J. Spencer, J. Huang, S. Ganguli, M. Sahami, L. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. volume 3, pages pAags. 19–23, 2015.
- [10] S. Reddy, S. Levine, and A. D. Dragan. Accelerating human learning with deep reinforcement learning. 2017.
- [11] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.
- [12] Y. Su, Q. Liu, Q. Liu, Z. Huang, Y. Yin, E. Chen, C. Ding, S. Wei, and G. Hu. Exercise-enhanced sequential modeling for student performance prediction. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [13] J. Zhang, X. Shi, I. King, and D.-Y. Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, pages 765–774. International World Wide Web Conferences Steering Committee, 2017.

Clustering Students Based on Their Prior Knowledge

Nisrine Ait Khayi
Institute for Intelligent Systems
The University of Memphis
Memphis, TN, 38152, USA
+19012473589
ntkhynyn@memphis.edu

Vasile Rus
Institute for Intelligent Systems
The University of Memphis
Memphis, TN, 38152, USA
+19016785259
vrus@memphis.edu

ABSTRACT

In this paper, we applied a number of clustering algorithms on pre-test data collected from 264 high-school students. Students took the pre-test at the beginning of a 5-week experiment in which they interacted with an intelligent tutoring system. The primary goal of this work is to identify clusters of students exhibiting similar knowledge patterns. In particular, we show that the DP-means clustering algorithm yields very good results using binary response data. Other clustering algorithms such as k-modes have demonstrated better results when using categorical response data.

Keywords

Clustering, intelligent tutoring systems, students modelling.

1. INTRODUCTION

Assessment is a key element in education in general and in Intelligent Tutoring Systems (ITSs) in particular because fully adaptive tutoring presupposes accurate assessment [2, 12].

Capturing students' knowledge state, our focus, and other learner characteristics that are important for learning such as their emotional state is critical to facilitate learning through adaptivity, i.e., tailoring instruction to each individual learner [11]. It should be noted that adaptivity can be thought of at two levels: macro-adaptivity which means selecting appropriate instructional tasks and micro-adaptivity which implies offering appropriate scaffolding while students work on a task (also called within-task adaptivity). Our work presented here could inform both micro- and macro-adaptivity. For instance, understanding the knowledge gaps of students in a particular cluster could inform what instructional tasks to choose for these students, i.e., it informs macro-adaptivity.

Indeed, an important preliminary step in creating an ITS that is sensitive to student misconceptions and individual learning trajectories is to first understand the various levels of mastery with respect to a target domain, for instance, physics. For example, important questions that need to be answered are: What are the predominant misconceptions they hold? and Are these misconceptions evenly distributed across topics and level of course taught? Using the clustering method proposed here will help

answer such important questions. To this end, in this paper, we document for each group of students identified by our clustering algorithm, the major misconceptions exhibited by that group.

In this study, we applied clustering on a pretest data collected at the beginning of an experiment in which high-school students interacted with a dialogue-based ITS. Our goal was to identify student groups and analyze them as a group in terms of misconceptions and mastered concepts. The identified groups could then be used to inform the authoring of instructional tasks and within-task instructional strategies and feedback for each group as opposed to each learner, which would be a much more expensive process. Learning such individualized strategies for each learner would be possible using automated methods, such as reinforcement learning, but they require substantially more experimental data which it is not have available.

The main clustering algorithm used in this study is the DP-means algorithm [4]. Its main advantage is identifying the number of clusters using a Dirichlet Process Mixture Model. After briefly presented related work and the context of our own work, what follows is a description of the DP-means algorithm. We then present details of the experiments and results. We conducted experiments using two types of data: binary and categorical responses. In addition, other clustering algorithms were employed to compare the results with those obtained with DP-means. We evaluated the performance of the resulted clusters using intrinsic, e.g., based on the silhouette index which measures the compactness of each cluster, and extrinsic methods, e.g., based on students' post-test scores derived from post-test responses which were not used to generate the clusters.

2. RELATED WORK

Clustering has been used in the past for analyzing education data as indicated by the research studies presented next. Bouchet and colleagues [1] have applied the Expectation-Maximization clustering algorithm on data collected from the MetaTutor ITS. MetaTutor scaffolds student's metacognitive skills while learning about the human circulatory system. The main objective of their clustering was reinforcing self-regulated learning via student profiling. The results consisted of three distinct clusters of students in terms of performance. The results have been analyzed using a MANOVA approach.

Rodrigo and colleagues [8] have applied k-means clustering on data collected from students interacting with Aplusix, an ITS for Algebra. The main research goal of their work was identifying students' behaviors through an analysis of interaction logs. The results have demonstrated the existence of two clusters of students

Nisrine Ait Khayi and Vasile Rus "Clustering Students Based on Their Prior Knowledge" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 246 - 251

associated with differing behavior and affective states. The first cluster reflected more collaborative work whereas the second cluster reflected more solitary work.

Reyes-Gonzalez and colleagues [7] have used the LC-Conceptual clustering algorithm from logical combinatorial pattern recognition for student modeling in an ITS. This algorithm is based on two phases: the first phase consists of building groups of objects based on their similarity and a grouping criterion. The second phase is called the intentional structure phase where the distinctive features of each resulted cluster are determined. Fang and colleagues [3] have used k-means clustering to capture learning patterns in over 250 students who used AutoTutor to gain reading comprehension skills. The average response times per question and performance across lessons have been used to cluster the students' learning behavior. The results showed the convergence of four types of learners: proficient readers, struggling readers, conscientious readers and disengaged readers. Classifying readers can improve the adaptivity of AutoTutor ITS by providing a proactive feedback and intervention based on the learning behaviors.

Similar to those other approaches, our intention was to discover groups of students with similar knowledge states as characterized by their responses to the multiple-choice pre-test. Each incorrect choice in the pre-test is associated with a major misconception and therefore students that pick similar choices should be assigned to the same cluster. The centroid of the cluster could then be used to interpret the strengths and weaknesses of students in that cluster and appropriate interventions designed for that group.

3. CONTEXT OF THIS WORK

Our work was conducted in the context of an experiment in which high-school students interacted with a dialogue-based intelligent tutoring systems that tutors students on science topics through problem-solving. The system encourages students to self-explain solutions to complex science problems and only offers help, in the form of hints, when needed, e.g., when the student is floundering. That is, during a typical tutorial session, the system challenges students to solve a number of problems that are carefully selected by the system in order to optimize student learning (macro-adaptivity). When working on a particular problem, students are first asked to provide a solution that must include a justification based on concepts and principles of the target domain, which was Newtonian Physics in the case of our study presented here. All other things equal, low knowledge students will most likely struggle to provide solid self-explanations and most likely to articulate misconceptions which would lead to more scaffolding dialogue moves in terms of hints and correcting misconceptions on the part of the computer tutor (micro-adaptation). High knowledge students would need less scaffolding and therefore the corresponding dialogues should be shorter.

Before students start interacting with the system, they took a pre-test to assess their initial knowledge state. The tool elected to assess students' initial knowledge state was an enhanced version of the Force Concept Inventory (FCI). The Force Concept Inventory (FCI) is a 30-item multiple-choice "test" designed to assess student understanding of the most basic concepts in Newtonian mechanics (Halloun, Hake, and Mosca, 1995). The FCI presents students with various situations and ask them to choose between Newtonian explanations for the phenomena, versus common-sense alternatives (Hestenes, Wells, & Swackhamer, 1992). The FCI has been widely used to measure learning in introductory physics courses. For example, Hake (1998) reported FCI data from 6,000 high school

and university students. Coletta and Phillips (2005) combined their data with data collected by Hake (1998) and in combination used the FCI to measure learning in 73 university and college introductory physics classes. The data we have is based on an augmented version of the FCI consisting of 35 multiple-choice questions. The augmented FCI adds a number of questions for certain Newtonian topics which were not covered enough in the original FCI test.

We administered the augmented Force Concept Inventory (aFCI) to students at three public and two private high schools in the mid-south region, including six teachers and 26 classrooms. The pretest was administered in classroom. Students completed the aFCI via provided scantron sheets, which were then collated and processed. The results of the scantron sheets were then compared to direct markings on the actual aFCI test in the case of blank or unidentifiable scantron responses. The data collection process was quite successful, resulting in 444 students with complete pretest data. We only used a subset of 265 students in our experiments because post-test data, used for extrinsic evaluation of our clustering, was available only for those 265 (the rest of the students either missed a tutoring session, or the post-test, or both).

It should be noted that the data is very diverse in terms of student prior knowledge of physics because students were recruited from a large variety of physics-related courses including introduction to physics, honors physics, and AP physics. This should allow us to draw general conclusions

4. DP-means BASED CLUSTERING

The DP-means algorithm, as described by Kulis & Jordan [4], is a hard-clustering approximation of nonparametric Bayesian models. Under the assumption that the DP-means is derived from a Dirichlet Process Mixture Model, there exists a lambda value α such that when used by the algorithm, the number of clusters k is identified. The DP-means algorithm is similar to the k-means clustering algorithm except that a new cluster is generated when the distance from a data point to the nearest cluster is larger than the threshold α .

Specifically, the DP-means algorithm is derived from a Dirichlet Process Mixture Model (DPMM) as illustrated below:

- $\mu_1, \dots, \mu_k \sim G_0$
- $\pi \sim \text{Dir}(k, \pi_0)$
- $z_1, \dots, z_n \sim \text{Discrete}(\pi)$
- $x_1, \dots, x_n \sim N(\mu_{z_i}, \sigma I)$
- The Dirichlet prior of dim k is placed using some π_0

where:

- μ is the mean of each of the clusters, drawn from some base distribution G_0 , which is the prior distribution over the means.
- $\pi = (\pi_1, \pi_2, \dots)$ corresponds to the vector of probabilities of being in a cluster.
- z_i is an indicator of cluster assignment.
- x_i is a data point

The corresponding clustering algorithm is described in Figure 1. The input consists of data instances x_1, \dots, x_n , where x_i represents the vector of pre-test answer choices of the i^{th} student. Since the pre-test contains 35 questions, each such response vector x_i contains 35 entries corresponding to each answer choice picked by

student i . The clustering algorithm begins by initializing a single cluster whose mean is the global centroid. Then, it initializes a set of cluster indicators: $z_i = 1$ for all $i = 1, \dots, n$ where $z_i = k$ means that the student x_i belongs to the k^{th} cluster as denoted by l_k .

In step 3, the algorithm computes the distances between each data point and the existing centroids. It then compares the minimum of these distances with α . If the minimum is larger than the threshold α , a new cluster is generated, and its centroid is assigned the current data point x_i . Otherwise, the cluster indicator of the current data point is set to the $argmin$ of the distances. After looping over all data points, the number of clusters k and the clusters indicators are computed. Finally, the DP-means algorithm generates the clusters l_j and their centroids μ_j for $j = 1, \dots, k$. Step 3 is repeated until the algorithm converges.

Algorithm: DP-means
Input: x_1, \dots, x_n : input data, α : cluster penalty parameter
Output: Clustering l_1, \dots, l_k and number of clusters k
1. Init. $k = 1, l_1 = \{x_1, \dots, x_n\}$ and μ_1 the global mean.
2. Init. Cluster indicators $z_i = 1$ for all $i = 1, \dots, n$
3. Repeat until convergence
• For each point x_i
- Compute $d_{ic} = \ x_i - \mu_c\ ^2$ for $c = 1, \dots, k$
- If $d_{ic} > \alpha$, set $k = k + 1, z_i = k$, and $\mu_k = x_i$
- Otherwise, set $z_i = argmin_c d_{ic}$
• Generate clusters l_1, \dots, l_k based on z_1, \dots, z_k :
$l_j = \{x_i \mid z_i = j\}$
• For each cluster l_j , compute $\mu_j = \frac{1}{ l_j } \sum_{x \in l_j} x$.

Figure 1. DP-means algorithm

4. EXPERIMENTS AND RESULTS

4.1 Dataset

The data used in our experiments consists of pre-test answers collected from 264 high-school students who took the aFCI pre-test, went through a 5-week training period, and then took a post-test. Furthermore, after each training sessions students took a short post-test (6 questions). In all our experiments, we will use this post-test after the very first training session as the extrinsic evaluation criterion as it is closest in time (among all post-tests) to the pre-test and therefore is a good estimate of students' early knowledge states as best captured by the pre-test. The pretest includes 35 multiple choice questions that have the same weight. Two types of data have been used in our experiments: 5-way response data and binary response data. The categorical data consists of the actual answer choices students picked for the 35 multiple choice questions coded as A, B, C, D and E. For each question, one those choices is the correct answer. The binary data represents the same data coded as binary correctness values: 0 – incorrect, i.e., the student picked any of the incorrect answer choices, and 1 – correct, i.e., the student picked the correct answer choice.

Tables 1 and 2 illustrate the data representation for the two tables. As described below, the columns represent the 35 questions and the rows represent individual students' responses.

Table 1. Categorical data

	Q1	Q2	...	Q35
Dh001	A	B	...	C
Dh002	C	D	...	C
...
DH356	C	D	...	B

Table 2. Binary data

	Q1	Q2	...	Q35
Dh001	0	0	...	0
Dh002	1	0	...	1
...
DH356	1	0	...	0

4.2 Experiments: Binary data

A first set of experiments have been conducted using the binary data as input for the DP-means algorithm. Since we have binary data and DP-mean is based on the Euclidean distance, we have applied Principal Component Analysis (PCA) to convert the binary values to continuous ones. For this purpose, numerous values of n (number of components) have been tested. The value 35 led to a convergence state of 10 clusters in which several clusters are redundant, i.e., using the extrinsic criterion based on the overall post-test score. For example, the average of the post test score for clusters 6, 7 and 9 is 3.0. Thus, we have tested randomly several values. The value 24 led to better clustering results in terms of splitting well the clusters based on the extrinsic criterion. Thus, we used those components to represent our data points for the rest of the experiments. On the binary data, a Manhattan distance could be used which we tried and didn't lead to better results than the above method of using PCA.

The α distance parameter has not been defined a priori. To select a suitable value of this parameter, we followed first the procedure described by Kulis and colleagues [4] as in the following: given $k=3$ as the desired number of clusters, we first initialize a set A with the global mean of the data. Then iteratively we calculate the maximum distance to A (the distance to A is the smallest distance among points in A). We repeat this $k (=3)$ times and assign to α the value of the maximum distance to A. In our work, we got the value of 3.26. Testing the DP-means with this value led to the convergence of two clusters of students. To reach the desired number of clusters which is 3, we have tried other values in a close interval of [3.26, 2.8] as described in Table 3.

Thus, various values have been tested and compared. The evaluation of the resulted clusters has been done using the following measures:

- Silhouette index: Its value measures how similar a student response vector (her set of responses to the pre-test questions) is to its own cluster (cohesion) relative to the other clusters (separation). The silhouette index is a value within the $[-1, +1]$ interval. A high value of the silhouette index indicates that the student is well matched with the other students in the same cluster. The following metric distances have been tested:

Euclidean distance, Manhattan distance and cosine similarity. The obtained results have shown that the Euclidean distance led to better results as demonstrated in Table 3.

- Mean of post test score: The data collected from the interaction of the students with the ITS includes post test scores for the 264 students. Since the post test is taken at the end of the experiment, weeks after the students took the pretest, and since it has not been used in the cluster, it can be used as an extrinsic measure of cluster validity and interpretation of the resulting clusters. Indeed, this measure is used by us to assess the mastery level of each resulted cluster of students. In addition, it has been used as a way to check the separation of the clusters. The maximum and minimum values of the post test score in this collected data are 6.0 and 0.0 respectively.
- Mean of pretest score: The data collected includes the pretest performance of each student based on of the correct answers. The highest value is 35 and the lowest value is 0.

Table 3 offers a set of results of DP-means clustering using different types of distances.

Table 3. clustering results with different types of distances

Distance	α	Number of clusters
Manhattan	2.9	255
	3.0	255
	3.1	255
Euclidian	2.9	5
	3.0	3
	3.1	2
Cosine	2.9	1
	3.0	1
	3.1	1

Table 4. DP-means clustering results with different values of α

α	Clusters	Mean pretest score	Mean post-test score	Number of students
2.9	C1	15.26	3.31	207
	C2	31.28	5.66	36
	C3	6.47	1.89	19
	C4	25.0	5.0	1
	C5	14.0	2.0	1
3.0	C1	17.68	3.44	195
	C2	31.11	5.64	37
	C3	8.83	1.62	32
3.1	C1	13.87	3.18	277
	C2	29.54	5.64	37

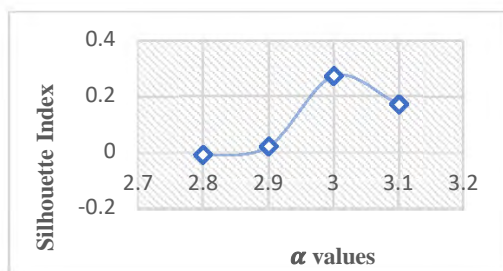


Figure 2. Quality of the DP-means algorithm using different values of α

The results in Figure 2 show that the value 3.0 of parameter α led to the highest value of the Silhouette index (0.27). In addition, this α value resulted in three distinct clusters, well separated (Figure 3), in terms of students' performance in the post test and pretest (as described in table 4). The first cluster contains 195 students. The mean post test score is 3.44 and the mean pretest score is 17.68 which are average scores. Students who belong to this cluster can be described as average performers or learners. The second cluster contains 37 students. The mean post test score is 5.66 and the mean pretest score is 31.11 which are high scores. The students in this cluster can be described as high performers or learners of Physics. The third cluster contains 32 students. The mean post test score is 1.625 and the mean pretest score is 8.83 which are very low scores. The students of this cluster can be describing as struggling ones.

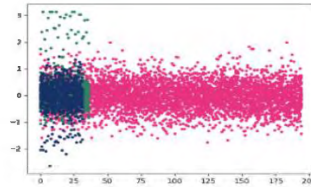


Figure 3. DP-means visualization with $\alpha = 3$

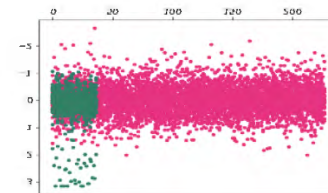


Figure 4. DP-means visualization with $\alpha = 3.1$

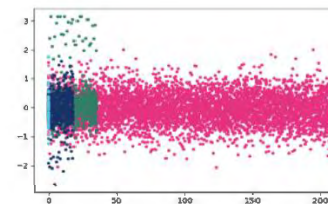


Figure 5. DP-means visualization with $\alpha = 3.2$

To compare the results of the DP-means algorithm with other clustering algorithms, we have also applied the k-means and agglomerative clustering algorithms on the same binary data. Since the best result of the DP-means was for an α value 3.0, we ran the k-means algorithm using $k=3$ and the agglomerative clustering using the same number of clusters ($=3$). Tables 5 and 6 present the results for k-means and agglomerative clustering, respectively.

Table 5. k-means results

Clusters	Mean Post-Test Score	Mean Pretest Score	Number of students
C1	2.28	9.44	97
C2	5.21	28.84	52
C3	3.83	17.22	115

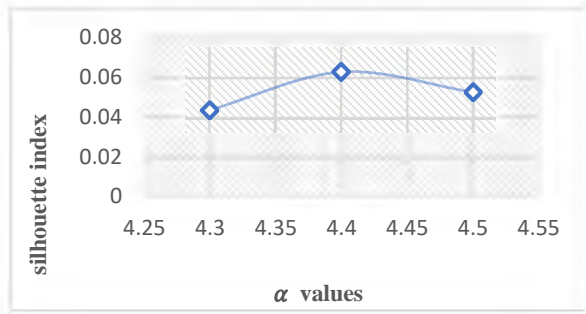


Figure 6. Quality of the DP-means algorithm using different values of α .

Table 6. Agglomerative clustering results

Clusters	Mean Post Test Score	Mean Pretest Score	Number of students
C1	3.55	16.46	165
C2	5.45	30.0	42
C3	2.07	8.28	57

The results depicted in Table 4 show that the DP-means algorithm with $\alpha=3.0$ outperforms the k-means and the agglomerative algorithms as described in tables 5 and 6 respectively. The difference, in the mean post test score and the mean of pretest score, between the clusters of DP-means is larger than the difference using the other clustering algorithms. This indicates the convergence of well separated groups of students, in terms of learning level and prior knowledge, when applying the DP-means.

A detailed analysis of the top 10 students closest to the centroids of each of the three clusters found by DP-means, revealed that students in cluster 1 struggled mostly with questions related to Newton's third and first laws, whereas students in cluster two struggled with questions related to Newton's third law. Students in cluster 3 struggled the most and they showed weaknesses across all major topics in Newtonian Physics. Since in this experiment we used just correctness values for each pre-test question it is not possible to provide a more detailed analysis in terms of specific misconceptions, e.g., assuming faster velocity implies a larger force in an action-reaction pair, students in each clusters exhibit.

4.4 Experiments: Clustering categorical data

A second set of experiments have been conducted using categorical data and the DP-means and K-modes clustering algorithms. That is, in this case, we used the actual answer choices picked by students for the pre-test questions in order to find the clusters.

To this end, first, we have converted the categorical responses to numerical ones using one-hot encoding. Basically, each answer choice becomes a dimension in a vector space representation. A value of 1 is assigned to that dimension for a given question in the pre-test if a student picked the choice corresponding to the dimension as their answer choice. This results in an encoding of categorical integer features as a one-hot numeric array. The encoder derives the categories based on the unique values in each feature. The output of the one-hot-encoding is fed into the clustering algorithms.

The results presented in Table 7 reflect a decrease in quality of the DP-means clustering using categorical data. The silhouette index, as described in Figure 6, has decreased in comparison with the DP-means based on binary data. The highest value was 0.06 when using the value 4.4 of α . The different values of α didn't lead to a good split of students in terms of the performance. For example, in the case of $\alpha = 4.4$, cluster C2 and C3 can be merged in one cluster since their mean post test and pretest scores are very close. For $\alpha = 4.3$, there is redundancy in the resulted clusters. For example, C3 and C6 can be merged in one cluster.

Table 7. DP-means clustering results (categorical data) with different values of α

α	Clusters	Mean Post Test Score	Mean Pretest Score	Number of students
4.3	C1	4.17	20.39	68
	C2	1.5	5.5	67
	C3	1.0	10.0	1
	C4	2.0	4.5	54
	C5	2.0	6.0	40
	C6	1.66	6.0	22
	C7	2.16	9.35	12
4.4	C1	4.12	20.14	187
	C2	1.0	10.0	1
	C3	1.66	6.0	3
	C4	2.15	9.0	73
4.5	C1	3.55	16.87	263
	C2	1.0	10.0	1

In order to overcome this drawback of DP-means when applied to categorical data, we have applied the k-modes clustering algorithm [5, 6]. The k-modes algorithm is based on defining the dissimilarity measure between objects. This dissimilarity between two objects A and B can be defined by the total mismatches of the corresponding attribute categories of the two objects. The smaller the number of mismatches is, the more similar the two objects. The dissimilarity measure is calculated using the following equation:

$$d(X, Y) = \sum_{j=1}^m \delta(x_j, y_j) \quad (1)$$

where:

$$\delta(x_j, y_j) = \begin{cases} 0 & (x_j = y_j) \\ 1 & (x_j \neq y_j) \end{cases} \quad (2)$$

The following are the results with k-modes when using $k = 3$.

Table 8. Kmodes results with $k = 3$

Clusters	Mean Post Test Score	Number of students
C1	3.72	120
C2	5.03	59
C3	2.23	85

The results listed in Table 8 demonstrate that the k-modes outperforms the DP-means when using categorical data. The

resulted clusters reflect a good split between clusters in terms of performance in the post test. The C1 cluster reflects an average knowledge level of students. C2 reflects a high level of knowledge of students. And C3 reflects a low level of learning. A more detailed analysis indicates the same overall conclusions reached using the correctness data, e.g., students in cluster one struggle mostly with Newton's second and third laws. However, when using the categorical data, we can further pinpoint which aspects of Newton's third law for instance, students struggle with. For instance, many students in cluster C1 seem to struggle with the misconception that in an interaction between two objects the more massive one will act with a bigger force on the smaller one which is not true. According to Newton's third law, to each action there is an equal and opposite reaction. Therefore, this analysis suggests that when a new student uses a Physics ITS, after they take the pre-test and their answer patterns place him closer to the centroid of cluster C1, i.e., in cluster C1, then appropriate instructional tasks that have been designed for students in that cluster should be activated in order to overcome major gaps students in that cluster exhibit.

5. CONCLUSIONS

In this work, DP-means clustering algorithm has been applied on the pretest data of 264 students collected from their interaction with DeepTutor ITS. Various values of α have been tested. The results demonstrated that 3.0 is the best value and three distinct clusters of students have been converged. These clusters reflect three distinct levels of learning which has been assessed using post test scores. The first cluster of students correspond to an average level of learning, the second cluster represents students with a high level of learning and the third cluster of students those with a low level of learning. Results have demonstrated also that DP-means outperforms k-means and Agglomerative clustering in terms of splitting well students based on their performance in the post test. Another finding is that the quality of DP-means algorithm, measured by the silhouette index, decreases when we use the categorical data in comparison with the binary data. To overcome this drawback, we have used the k-modes clustering.

Furthermore, such clustering could offer a good trade-off between adaptivity and authoring costs. For instance, macro-adaptation can be expensive if the number of unique student knowledge states is very large as it requires selecting a unique set of tasks for each such unique knowledge state. Concretely, if using a 5-way/choice 35 multiple-choice question pre-test, the number of possible combinations of 35 answers is 5^{35} , an extremely large space. That is, if each student's knowledge state is described by the 35 responses we end up with 5^{35} student knowledge states or student models which, by comparison, is much larger than the world's population which is a bit over 5^{14} . Considering each of these potential knowledge states and selecting for each corresponding learner a unique set of tasks becomes a computationally and authoring challenge. An alternative, for instance, would be to group students into clusters of similar mental models and then select and author tasks for each such clusters. That is, grouping students into similar mental model groups can offer a good trade-off between adaptivity and authoring costs.

We plan to further investigate the resulting clusters for a better understanding of the characteristics of the students in each cluster. For instance, we do have information about the Physics class (intro, honors, AP) each student took and therefore a detailed analysis for students in each cluster based on their class type can be performed in order to understand what are the major misconceptions students

in each class struggle with. Not only this could inform an ITS for Physics, but this information can be shared with teachers in order to help them better plan their lessons plans to address major misconceptions their students may have.

6. ACKNOWLEDGMENTS

This research was partially sponsored by the University of Memphis and the Institute for Education Sciences under award R305A100875 to Dr. Vasile Rus.

7. REFERENCES

- [1] Bouchet, F., Harley, J. & Trevors, G. & Azevedo, R. (2013). Clustering and Profiling Students According to their Interactions with an Intelligent Tutoring System Fostering Self-Regulated Learning. *Journal of Educational Data Mining*, 5, 104-146.
- [2] Chi, M. T. H., Siler, S., Jeong, H., Yamauchi, T., & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science*, 25, 471-533 (2010)
- [3] Fang, Y., Shubeck, K., Lippert, A. & Cheng, Q. & Shi, G. & Feng, S. & Gatewood, J. & Chen, S. & Cai, Z., Pavlik, Jr, P., Frijters, J., Greenberg, D. & Graesser, A. (2018). Clustering the Learning Patterns of Adults with Low Literacy Skills Interacting with an Intelligent Tutoring System.
- [4] Kulis, B. & Jordan, M. I. (2012). Revisiting k-means: New algorithms via Bayesian nonparametrics. *In Proceedings of the 23rd International Conference on Machine Learning*, 2012.
- [5] Huang, Z. (1997). Clustering large data sets with mixed numeric and categorical values, *Proceedings of the First Pacific Asia Knowledge Discovery and Data Mining Conference*, Singapore, pp. 21-34, 1997.
- [6] Huang, Z. (1998). Extensions to the k-modes algorithm for clustering large data sets with categorical values, *Data Mining and Knowledge Discovery* 2(3), pp. 283-304, 1998.
- [7] Reyes-González, Y., Martínez-Sánchez, N., Díaz-Sardiñas, A. & Patterson-Peña, M.C. (2018). Conceptual clustering: A new approach to student modeling in Intelligent Tutoring Systems. *Revista Facultad de Ingeniería*. 70-76. 10.17533/udea.redin.n87a09.
- [8] Rodrigo, Ma.M.T., A. Anglo, E., O. Sugay, J., & Baker, R. (2008). Use of Unsupervised Clustering to Characterize Learner Behaviors and Affective States while Using an Intelligent Tutoring System. *Proceedings - ICCE 2008: 16th International Conference on Computers in Education*. 49-56.
- [9] Rus, V., D'Mello, S. K., Hu, X., & Graesser, A. C. (2013). *Recent advances in intelligent tutoring systems with conversational dialogue*, *AI Magazine*, 34(3), 42-54.
- [10] Rus, V., Niraula, N.B., and Banjade, R. (2015). DeepTutor: an effective, online intelligent tutoring system that promotes deep learning. *In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press 4294-4295.
- [11] Shute, V.J., Zapata-Rivera, D. (2012). Adaptive educational systems. *Adaptive technologies for training and education*, P. DURLACH, Ed. New York, NY: Cambridge University Press, 7-27.
- [12] Woolf, B.: *Building Intelligent Interactive Tutors, Student-Centered Strategies for Revolutionizing E-Learning*, Elsevier & Morgan Kaufmann Publishers, (2008).

Grades are not Normal: Improving Exam Score Models Using the Logit-Normal Distribution

Noah Arthurs
Stanford University
narthurs@cs.stanford.edu

Ben Stenhaus
Stanford University
stenhaus@stanford.edu

Sergey Karayev
Gradescope
sergeyk@gradescope.com

Chris Piech
Stanford University
piech@cs.stanford.edu

ABSTRACT

Understanding exam score distributions has implications for item response theory (IRT), grade curving, and downstream modeling tasks such as peer grading. Historically, grades have been assumed to be normally distributed, and to this day the normal is the ubiquitous choice for modeling exam scores. While this is a good assumption for tests comprised of equally-weighted dichotomous items, it breaks down on the highly polytomous domain of undergraduate-level exams. The logit-normal is a natural alternative because it has a bounded range, can represent asymmetric distributions, and lines up with IRT models that perform logistic transformations on normally distributed abilities. To tackle this question, we analyze an anonymized dataset from Gradescope consisting of over 4000 highly polytomous undergraduate exams. We show that the logit-normal better models this data without having more parameters than the normal. In addition, we propose a new continuous polytomous IRT model that reduces the number of item-parameters by using a logit-normal assumption at the item level.

1. INTRODUCTION

Historically, student performance on exams has been assumed to be normally distributed. Grade curving originates from the idea that students exist on a “bell curve,” in which most are clustered around the mean and a small number over- or under-achieve. The field of education has many criticisms for the bell-curve mindset. A common argument is that we should not take the *observation* that student performance tends to look normal and turn it into a normative practice [4, 23]. The idea that some students will inevitably fail and only a small number can enjoy the highest level of success runs counter to the goals of the educator, who should want as many students as possible to succeed. This tension plays out in the ideological battle between those who criticize grade inflation [9] and those who suggest that students may be earning the higher grades they are receiving [11].

Noah Arthurs, Ben Stenhaus, Sergey Karayev and Chris Piech "Grades are not Normal: Improving Exam Score Models Using the Logit-Normal Distribution" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 252 - 257

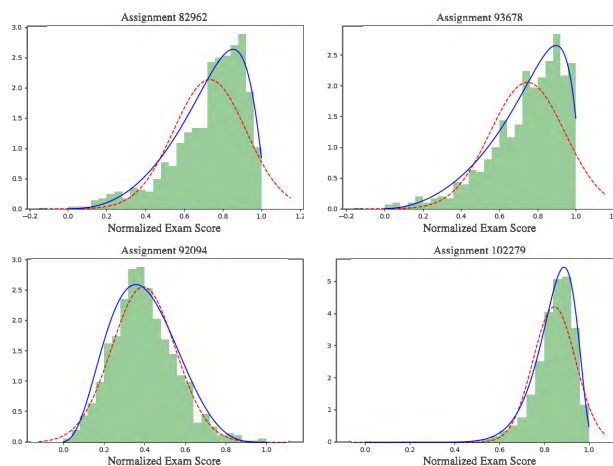


Figure 1: Score histograms of four assignments, along with the PDFs of the best-fit normals (dashed red) and best-fit logit-normals (solid blue).

The normal assumption is commonplace in modern research into educational data. Grade distributions are usually presented to students in terms of their mean and variance, and they are often visualized as normal distributions [17]. As education becomes more digitized, statistical models of grading become more widespread. For example, peer grading models allow MOOC’s to assign accurate grades to students based on noisy estimates from their classmates. State of the art peer grading models use normal priors over grades [19, 18], which will result in normal-looking distributions. Both of these examples can benefit from challenging the normal assumption. In the first case, finding new ways to parameterize grade distributions can help us better interpret and visualize student behavior. In the second, a more accurate prior over grades would help peer grading models assign more accurate grades to students.

In this paper, we analyze over 4000 assignments graded on the Gradescope platform [22]. These assignments are primarily exams from undergraduate STEM courses, and as a result the data is highly polytomous (many scoring categories per question). However, principal component analysis (PCA) reveals that these exams have good low-rank approximations, meaning that the data is very structured.

First, we examine the ability of different families of distributions to model the scores in our dataset. Specifically we compare the normal to three bounded two-parameter distributions. We find that the logit-normal [1, 5] is consistently the best choice, followed by the beta, which is known to approximate the logit-normal [8].

In the second part of this paper, we build a simple continuous polytomous IRT model using a logit-normal assumption at the item level. Our model outperforms both the Generalized Partial Credit Model [16] (a standard discrete IRT model) and the Continuous Response Model [20] (a standard continuous IRT model) on the Gradescope data, despite having fewer parameters than either. This indicates that we can simplify and improve polytomous IRT models using structural assumptions about assignment data.

1.1 Related Work

When analyzing student behavior, it can be difficult to distinguish between cases where data is actually normal and cases where an assumption of normality is influencing the distribution. For example, SAT scores are known to be normally distributed, but this is because raw SAT scores are translated into final scores using a system that enforces a normal distribution [3]. More subtly, probabilistic models for determining scores based on peer grades often use normal priors over their output [18, 19]. As a result, they will push grade distributions to be normal. The question then remains about whether these distributions *should* look normal in reality or another prior needs to be found.

Of course, grade curving is the most direct way in which student performance is influenced to be normal. Although it remains a common practice, research has shown that most students prefer not to be graded on a curve [6], that both students and professors find indiscriminate grade curving unethical [15], and that grade curving can amplify the randomness of test-taking as a measure of student aptitude [12]. It has also been argued that educators should be striving to avoid normally distributed student outcomes, rather than enforce them [4, 23]. If this is the case, then we need to actively seek out new distributions for describing and understanding test scores.

Polytomous IRT models generally fall into two categories. Discrete models like Generalized Partial Credit [16] and Graded Response Models [21] model each point on each question separately, scaling with the number of scoring categories per question. These models make very few assumptions about the relationship between different scores and thus do not take advantage of any underlying structure in the data. Continuous models like the Continuous Response Model [20] are used less frequently, but they scale only with the number of questions in the assignment. They do this by making assumptions about the structure of the item characteristic curves (ICC's). This means that if a dataset's ICC's follow a consistent pattern, then a continuous model can thrive.

The Gradescope data we are working with is much more polytomous than most IRT datasets. This is because it comes from a wide variety of college-level courses rather than standardized tests. Despite this heterogeneity, past work on Gradescope data has found patterns in question

ordering and the interpretation of the first several principal components [13]. This indicates that there may be underlying structure that a continuous IRT model could take advantage of.

2. THE DATASET

Our initial dataset consists of 6,607 assignments submitted to Gradescope, an online tool for uploading and grading student work [22]. Typically students will do their assignments by hand using a template provided by the instructor. After the assignments have been scanned and uploaded to Gradescope, instructors can grade them using a digital rubric that is always visible and modifiable. To ensure that the majority of the data consists of college-level exams, all included assignments:

- are instructor-uploaded¹
- have a fixed template
- have at least 3 questions
- have titles that do not include terms that describe other kinds of student work (e.g. “HW” or “Quiz”)
- have titles that do not include terms that are indicative of non-college-level work

The assignments were graded between Spring 2013 and Spring 2018, and come from 2748 courses at 139 different higher-education institutions². The top three subject areas in the dataset are Electrical Engineering & Computer Science (50% of assignments), Math & Statistics (22%), and Chemistry (11%). The median number of courses per school is 4, and the median number of assignments per course is 2.

When fitting curves to exam scores, we want there to be enough values that the distribution is interesting/nontrivial and enough data points that the observed histogram is somewhere near the underlying distribution. For this reason, we filter out all assignments that have fewer than 10 unique scores or fewer than 75 students. We are then left with 4115 assignments. Figure 2(b) shows the joint distribution between student count and number of unique scores, as well as their marginals. Observe that the vast majority of assignments have 75-200 students and 20-80 unique scores.

Throughout this study we store each assignment as a matrix A where A_{ij} represents student j 's score on question i . We use the term “exam score” to refer to the sum of a student's question scores, so to analyze exam scores, we sum the rows of the assignment matrix A .

2.1 Visualizing the Data

One shortcoming of the normal is that it can only represent symmetric distributions.³ We measure the symmetry of an exam score distribution using skew (skewness), which can be

¹On Gradescope, students tend to upload their own homework, while exams tend to be scanned and uploaded by the instructor.

²However, UC Berkeley, UC San Diego, Stanford University, University of Michigan, and University of Washington account for half of all assignments in the dataset.

³This may not be a problem on all forms of test data. For example, if questions were equally weighted and (relatively) independent, the Central Limit Theorem would predict a symmetric distribution.

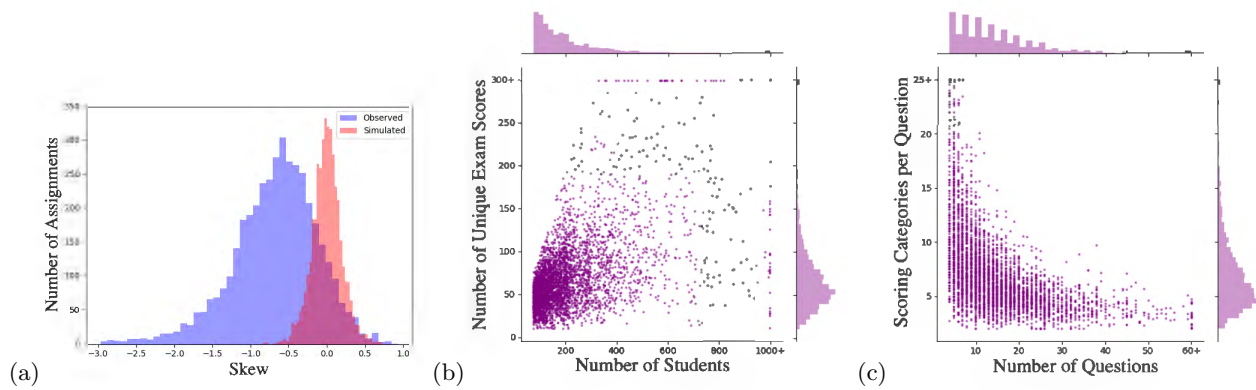


Figure 2: (a) Normal doesn’t fit. Blue histogram: assignment skew distribution. Red histogram: skews of assignments had they been drawn from normal distributions (as described in section 2.1). (b) Number of students vs. number of unique exam scores for each assignment in the filtered dataset. (c) Number of questions vs. average number of scoring categories per question for each assignment in the filtered dataset.

a good indicator of how normal a distribution is [2]. A skew value of (or near) 0 indicates a symmetric distribution, while negative and positive skew values indicate large tails on the left and right respectively. The blue histogram in Figure 2(a) shows the skews of the exam score distributions in our dataset. Note that they tend to be negative, meaning that exams tend to have larger tails of below-mean students than of above-mean students. To generate the red histogram in Figure 2(a), we performed the same experiment on a simulated dataset created by redrawing the scores of each assignment from its best-fit normal⁴. The large difference in both mean and variance of these histograms shows that our observed skews would not be very likely were the data normally distributed. In order to quantify this intuition, we performed a D’Agostino’s K-squared test of normality to determine how likely it would be for each assignment’s skew to arise from a normal distribution [2]. We found that 73% of assignments had a p-value of 0.05 or lower, indicating that (just on the basis of skew) the vast majority of assignments are very unlikely to have come from a normal distribution.

When fitting IRT models to the assignments, we are interested in how polytomous each assignment is. Figure 2(c) shows the joint distribution between the number of questions an assignment has and the average number of scoring categories per question. The negative correlation between these two stats is unsurprising⁵, but it means that we can test our models on highly polytomous items or large question-counts but not both at the same time.

2.2 Dimensionality

PCA [10] can give us insight into the dimensionality of our data. A previous use of PCA on Gradescope data [13] found that the first principal component distinguishes between high and low scoring students, while later principal components correspond to skill at particular types of questions (e.g. multiple choice, free response).

⁴If an assignment had n students, sample mean \bar{x} and sample variance S , our simulated version of that assignment would consist of n draws from $N(\bar{x}, S)$.

⁵There are only so many points that a student could be expected to earn over the course of a single exam.

We use PCA to describe the dimensionality of a given exam, where dimensionality is defined as the number of principal components required to account for 80% of the variance between students. Put simply, we are interested in what rank is required to form a “pretty good” approximation for the exam matrix. Intuitively, if exams have low dimensionality, then they have a large amount of structure we can exploit when modeling them.

We find that number of students and number of questions (the two dimensions of our exam matrix) do not influence dimensionality in the same way. Number of students is weakly correlated with dimensionality (0.20 Pearson correlation), and on average it requires over 250 extra students to add a dimension. Number of questions, on the other hand is more strongly correlated (0.85 Pearson correlation), and one dimension is added every 3.3 questions. This is consistent with the finding in [13] that principal components correspond to specific student aptitudes. Overall, this analysis indicates that choice of model should be based on the features of the exam itself, not on how many students are taking it.

We also examined the first principal component in isolation, and found that it generally indicated a student’s score. Across our dataset, the average magnitude of the Pearson correlation between exam score and the first principal component was 0.965. In addition, the first principal component on average accounts for 43% of the variability in an assignment. The strength of the first principal component indicates that IRT models might only need one-dimensional student ability parameters to be successful on this dataset.

3. FITTING EXAM SCORES

Our first modeling task is to compare the ability of different two-parameter distributions to fit the exam scores in our dataset. Since tests have minimum and maximum scores, we choose bounded distributions. In addition, due to our findings in 2.1, we choose distributions that are not symmetric.

3.1 The Distributions

The truncated normal distribution is the result of bounding the normal above and below. It is characterized by the

	Normal	Trunc	Beta	Logit
Beats Normal	-	100%	92%	87%
Beats Trunc	0%	-	67%	75%
Beats Beta	8%	33%	-	68%
Beats Logit	13%	25%	32%	-
Average LL	0.272	0.333	0.336	0.353

Table 1: Win Rates and Likelihoods: How often does each distribution outperform the others? The logit-normal, beta and truncated normal models are all better replacements for the normal distribution. logit-normal has the highest likelihood.

mean and variance of its underlying normal, and when the bounds are known, there is a closed form maximum likelihood (MLE) estimate of these parameters [7]. The truncated normal assigns a higher probability density to every value in its domain than its underlying normal does, and as a result it will strictly outperform the normal distribution in likelihood. Although it is not symmetric around its mean, if it includes the mean of the underlying normal, then its probability density function (PDF) will be mirrored across that point. The truncated normal will be a good fit if test scores are drawn not from a normal distribution but from a slice of a normal distribution.

The beta distribution is the conjugate prior of the Bernoulli, characterized by two parameters referred to as α and β . It has no closed form MLE estimates, but there is a closed form method of moments solution that can be used as a starting point for optimization. When $\alpha = \beta$, the beta has no skew, but it can achieve a wide range of skew values by varying the difference between the two parameters. The beta does not have an intuitive interpretation in this context.

The logit-normal distribution [1, 5] is the result of applying the sigmoid (logistic) function⁶ to data sampled from a normal distribution. Like the truncated normal, its parameters are the mean and variance of its underlying normal. The logit-normal and beta are known to approximate each other [8], but the logit-normal comes with the advantage of having closed form MLE estimates of its parameters⁷. It also has a nice interpretation that comes from item response theory. Logistic IRT models like 1PL/2PL/3PL take normally-distributed student abilities and use a linear transformation plus a sigmoid to transform them into probabilities. If the logit-normal is a good fit for exam scores, we can see it as performing the same kind of transformation to go from an underlying unbounded variable to an observed bounded one.

In addition to these, we tried a Two Gaussian Mixture Model in case distributions were bimodal. However, it performed worse than all three of these distributions despite having more parameters, so we did not include it in our results.

⁶The sigmoid, given by $\sigma(x) = \frac{1}{1+e^{-x}}$ compresses the reals into the range (0, 1). Its inverse, the logit function given by $\sigma^{-1}(p) = \log \frac{p}{1-p}$, does the opposite.

⁷Unsurprisingly, the MLE estimates for the mean and variance of the underlying normal are the mean and variance of taking the logit (σ^{-1}) of the exam scores.

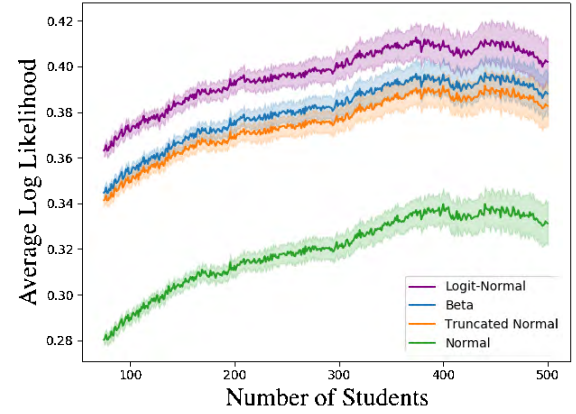


Figure 3: Difference in performance between the candidate distributions across sample sizes. Bands show standard error. Assignments were downsampled to simulate lower student counts.

3.2 Evaluating the Distributions

We fit each of the three distributions above plus the normal to each of the 4115 sets of exam scores in our filtered dataset. In order to more easily fit our bounded distributions to the data, we compress all scores into the range [0.05, 0.95]⁸. Specifically, if we have observed exam scores x_1, \dots, x_N , we map each x_i to

$$x'_i = 0.9 * \frac{x_i - x^{min}}{x^{max} - x^{min}} + 0.05$$

where $x^{min} = \min_i x_i$ and $x^{max} = \max_i x_i$. We use MLE parameter estimation to fit the distributions and evaluate them using log likelihood, defined as

$$LL(\theta) = \frac{1}{N} \sum_{i=1}^N \log f(x'_i | \theta)$$

where f is a PDF parameterized by θ^9 . We obtain the same semantic results when we use Earth Mover's Distance instead of Log-Likelihood as our goodness of fit metric.

3.3 Distribution Results

After performing this experiment, we find a clear hierarchy with the logit-normal performing best, followed by the beta, then the truncated normal, then the normal. Table 1 shows this hierarchy in two ways. First, the average log likelihood across assignments increases from left to right. Second, we can see that the logit-normal is a better fit than the beta 67% of the time, the beta is a better fit than the truncated normal 67% of the time, and the truncated normal is a better fit than the normal 100% of the time¹⁰. It is a little bit surprising that the beta outperforms the normal slightly more often

⁸[0, 1] may seem like the more natural choice, but both the beta and the logit-normal perform poorly when values are close to 0 or 1 (with the logit-normal unable to produce 0's and 1's at all).

⁹Note that because the PDF's are constrained to the range 0 to 1, our log likelihoods will come out positive.

¹⁰As mentioned above, this is because the truncated normal's PDF lies strictly above the PDF of its underlying normal.

	Baseline	GPCM	CRM	LNM
Parameters/Item	0	B + 1	3	2
Average RMSE	0.307	0.258	0.278	0.255

Table 2: Loss for each IRT model measured across 4115 assignments. “B” refers to the number of scoring categories for a given item.

than the truncated normal does, but this is the only result in Table 1 that is inconsistent with our proposed hierarchy.

Figure 3 shows that our conclusion that the logit-normal is the best choice is robust to sample size. All distributions fit better as sample size increases, since larger numbers of students result in smoother score histograms.

4. LOGIT-NORMAL IRT

Many of our results so far indicate that highly polytomous exam data is also highly structured. Polytomous IRT schemes like the Generalized Partial Credit Model (GPCM) [16] and Graded Response [21] scale with the number of scoring categories per question and thus do not take advantage of structure in the shapes of the item characteristic curves. On the other hand, continuous models like the Continuous Response Model (CRM) are able to cut down on parameters by assuming a parameterized function for each ICC. In this final section, we will propose a continuous model that uses logit-normals to make such simplifying assumptions and thus take advantage of the underlying structure of our data.

We find that when it comes to fitting exam scores, the logit-normal is just as successful when there are smaller numbers of unique scores available. Our model pushes that idea to its limit by modeling each question on an exam with a single logit-normal. The assumption is that highly polytomous items will behave like mini exams and as a result logit-normals will describe them well.

Our model fits a single ability $\theta_j \in \mathbb{R}$ to each student j and (as alluded to above) fits a logit-normal distribution to each item i with parameters μ_i and σ_i . Let S_i represent a random student’s score on question i , and let S_{ij} represent student j ’s score on question i . Our parameters are then related by the following equations:

$$\begin{aligned}\theta &\sim N(0, 1) \\ S_i &\sim \text{Logit-Normal}(\mu_i, \sigma_i) \\ E[S_{ij}] &= \sigma(\sigma_i \theta_j + \mu_i)\end{aligned}$$

As in section 2, we refer to our observed data as a matrix A where A_{ij} stores student j ’s score on question i . In addition, we assume that the data has been shifted and scaled as described in section 3.2. We fit this model in two steps:

1. Use MLE estimation to choose each μ_i and σ_i to fit the observed distribution of S_i ’s. Specifically, if A_i is the vector of observed scores on question i , we set μ_i and σ_i to be the sample mean and sample standard deviation of $\sigma^{-1}(A_i)$.¹¹

¹¹Here we are applying the logit function σ^{-1} element-wise.

2. Choose each θ_j to minimize the total squared error of the $E[S_{ij}]$ ’s. Specifically:

$$\begin{aligned}\theta_j &= \operatorname{argmin}_{\theta} \sum_i (E[S_{ij}] - A_{ij})^2 \\ &= \operatorname{argmin}_{\theta} \sum_i (\sigma(\sigma_i \theta_j + \mu_i) - A_{ij})^2\end{aligned}$$

We use least squares to fit the θ ’s because we have not defined a probability distribution over S_{ij} , which would be required to perform MLE. More research is required to determine what kind of probability distribution centered at $E[S_{ij}]$ will perform best.

4.1 Evaluating the IRT Models

We will evaluate our model based on how well it can use the parameters it has learned to predict student scores on each question. Note that in-sample evaluation is the norm for IRT [14]. Specifically we will measure the RMSE between the predicted $E[S_{ij}]$ ’s and the observed A_{ij} ’s. If an assignment has n students and m questions, this is calculated as:

$$RMSE(\theta, \mu, \sigma) = \left(\frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n (E[S_{ij}] - A_{ij})^2 \right)^{1/2}$$

4.2 IRT Results

Table 2 shows the comparison across the whole dataset between our Logit-Normal Model (LNM) and:

- a baseline that uses each student’s normalized exam score as a prediction for each question.
- a 2PL Generalized Partial Credit Model (GPCM) [16] fit using EM.
- a Continuous Response Model (CRM) [20] fit using the EM approach described in [24].

The fact that our model has the best performance despite having the fewest parameters indicates that it is taking advantage of the structure of highly polytomous items. We can conclude that discrete models are more complicated than necessary on data like this. We can also conclude that the assumptions about item characteristic curves in CRM are not as good as the logit-normal assumption on this data.

5. CONCLUSIONS

Overall, we have shown that highly polytomous exam data has a large amount of underlying structure that can help us simplify our probabilistic models. Out of three bounded, asymmetric candidates, the logit-normal came out on top as the best prior for exam scores. In addition, the logit-normal’s ability to model individual polytomous items allowed us to develop a polytomous IRT model that is simple and well-suited to this kind of data. We hope to have challenged the traditional assumption that the normal is the best prior for student behavior, and we hope that more work will be done to simplify IRT models for the highly polytomous data produced by college-level courses.

6. FUTURE WORK

The main loose end from this paper is the fact that we did not define full probability distributions over S_{ij} in the logit-normal model. Without these distributions, the model is

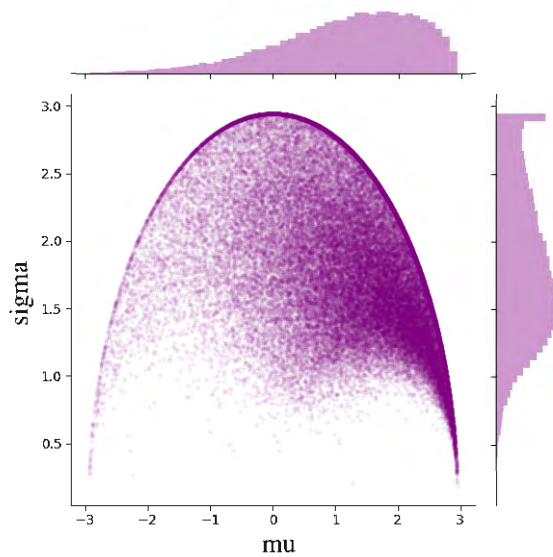


Figure 4: Joint distribution of σ and μ item parameters from the Logit-Normal Model.

able to predict scores and abilities but unable to act as a generative model. Future research needs to be done to find out what family of distributions best models S_{ij} . In addition, Figure 4 shows that there is a strong (somewhat ellipsoidal) relationship between the item parameters of our model, indicating that there may be further structure to exploit in highly polytomous items.

While our results are convincing in the highly-polytomous domain, more work is required to see how well they generalize to less polytomous data. In addition, the logit-normal needs to be tested on downstream tasks like peer grading in order to verify that it is an effective prior for exam scores.

In the interest of reproducibility, and to enable further science, the fully anonymized dataset used in this paper will be made available to other researchers for appropriate academic use. To gain access to the data, researchers must provide IRB approval documentation and must sign an agreement that ensures the anonymized data is treated appropriately.

7. REFERENCES

- [1] J. Atchison and S. M. Shen. Logistic-normal distributions: Some properties and uses. *Biometrika*, 67(2):261–272, 1980.
- [2] R. B. D’agostino, A. Belanger, and R. B. D’Agostino Jr. A suggestion for using powerful and informative tests of normality. *The American Statistician*, 44(4):316–321, 1990.
- [3] N. J. Dorans. Recentering and realigning the sat score distributions: How and why. *Journal of Educational Measurement*, 39(1):59–84, 2002.
- [4] L. Fendler and I. Muzaffar. The history of the bell curve: Sorting and the idea of normal. *Educational Theory*, 58(1):63–82, 2008.
- [5] P. Frederic and F. Lad. Two moments of the logitnormal distribution. *Communications in Statistics-Simulation and Computation*, 37(7):1263–1269, 2008.
- [6] J. F. Gaultney and A. Cann. Grade expectations. *Teaching of Psychology*, 28(2):84–87, 2001.
- [7] A. Hald. Maximum likelihood estimation of the parameters of a normal distribution which is truncated at a known point. *Scandinavian Actuarial Journal*, 1949(1):119–134, 1949.
- [8] N. L. Johnson. Systems of frequency curves generated by methods of translation. *Biometrika*, 36(1/2):149–176, 1949.
- [9] V. E. Johnson. *Grade inflation: A crisis in college education*. Springer Science & Business Media, 2006.
- [10] I. Jolliffe. *Principal component analysis*. Springer, 2011.
- [11] A. Kohn. The dangerous myth of grade inflation. *The Chronicle of Higher Education*, 49(11):B7, 2002.
- [12] G. Kulick and R. Wright. The impact of grading on the curve: A simulation analysis. *International Journal for the Scholarship of Teaching and Learning*, 2(2):n2, 2008.
- [13] P. Laskowski, S. Karayev, and M. A. Hearst. How do professors format exams?: an analysis of question variety at scale. 2018.
- [14] A. Maydeu-Olivares. Goodness-of-fit assessment of item response theory models. *Measurement: Interdisciplinary Research and Perspectives*, 11(3):71–101, 2013.
- [15] B. L. Morgan and A. J. Korschgen. The ethics of faculty behavior: Students’ and professors’ views. *College student journal*, 35(3):418, 2001.
- [16] E. Muraki. A generalized partial credit model: Application of an em algorithm. *ETS Research Report Series*, 1992(1):i–30, 1992.
- [17] R. O’Dea, M. Lagisz, M. Jennions, and S. Nakagawa. Gender differences in individual variation in academic grades fail to fit expected patterns for stem. *Nature communications*, 9(1):3777, 2018.
- [18] C. Piech, J. Huang, Z. Chen, C. Do, A. Ng, and D. Koller. Tuned models of peer assessment in moocs. *arXiv preprint arXiv:1307.2579*, 2013.
- [19] K. Raman and T. Joachims. Methods for ordinal peer grading. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1037–1046. ACM, 2014.
- [20] F. Samejima. Homogeneous case of the continuous response model. *Psychometrika*, 38(2):203–219, 1973.
- [21] F. Samejima. Graded response models. In *Handbook of Item Response Theory, Volume One*, pages 123–136. Chapman and Hall/CRC, 2016.
- [22] A. Singh, S. Karayev, K. Gutowski, and P. Abbeel. Gradescope: A fast, flexible, and fair system for scalable assessment of handwritten work. In *Proceedings of the fourth (2017) acm conference on learning@ scale*, pages 81–88. ACM, 2017.
- [23] R. J. Sternberg. The school bell and the bell curve. why they don’t mix. *NASSP Bulletin*, 80(577):46–56, 1996.
- [24] C. Zopluoglu. A comparison of two estimation algorithms for samejima’s continuous irt model. *Behavior research methods*, 45(1):54–64, 2013.

Assessing Student Response in Tutorial Dialogue Context using Probabilistic Soft Logic

Rajendra Banjade
The University of Memphis
Memphis, TN, USA
rajbanjade@gmail.com

Vasile Rus
The University of Memphis
Memphis, TN, USA
vrus@memphis.edu

ABSTRACT

Automatic answer assessment systems typically apply semantic similarity methods where student responses are compared with some reference answers in order to assess their correctness. But student responses in dialogue based tutoring systems are often grammatically and semantically incomplete and additional information (e.g., dialogue history) is needed to better assess their correctness. In that, we have proposed augmenting semantic similarity based models with, for example, knowledge level of the student and question difficulty and jointly modeled their complex interactions using Probabilistic Soft Logic (PSL). The results of the proposed PSL models to infer the correctness of the given answer on DT-Grade dataset show the more than 7% improvement on accuracy over the results obtained using a semantic similarity model.

Keywords

Tutoring System, Answer Assessment, Probabilistic Soft Logic

1. INTRODUCTION

Open ended answers are responses produced by students to questions, e.g. in a test or in the middle of a tutorial dialogue. Such answers are very different from answers to multiple choice questions where students just choose one or more options from the given choices and they are more easier to evaluate than open ended answers. In conversational Intelligent Tutoring Systems (ITSs; [18, 14]), the systems should be able to assess the students' responses in order to provide them appropriate feedback and to plan the subsequent part of the dialogue.

The true understanding of student answers is intractable as it requires collecting and doing reasoning over a huge knowledge, including the linguistic knowledge, domain knowledge, and world knowledge. As a practical alternative, semantic similarity methods are applied [5, 10, 15]. In this approach, systems assess student responses by measuring how much

of the targeted concept is present in the student answer. Accordingly, the subject matter experts create target (or reference) answers to the questions that students will be prompted to answer and the system assesses how much of the targeted concept is present in the student answer by measuring the semantic similarity between student's answer with reference answer.

The meaning of the reference answer is known because they are created by subject matter experts. The high similarity between student answer with reference answer indicates that the answer is correct. Otherwise, the answer is partially correct, or incorrect. This approach has been widely used in understanding student responses in tutoring systems and in automatic answer assessment systems in general (see Section 2). It is fast, does not require too much of information, and has been often found to be effective.

However, the implied assumption in similarity based answer assessment approach is that the student answer and the reference answer are self contained (i.e., grammatically and semantically complete). But student responses in conversational tutoring systems vary a lot as illustrated in Table 1. The meanings of students' responses often depend on the dialogue context and problem/task description. For example, students frequently use pronouns, such as *they*, *he*, *she*, and *it*, in their response to tutor's questions or other prompts. In an analysis of tutorial conversation logs, Niraula *et al.*[13] found that 68% of the pronouns used by students were referring to entities in the previous utterances or in the problem description. In addition to anaphora, complex coreferences are also employed by students.

Furthermore, in tutorial dialogues students react often with very short answers which are easily interpreted by human tutors as the dialogue context offers support to fill-in the blanks or untold parts. Such elliptical utterances are common in conversations even when the speakers are instructed to produce more syntactically and semantically complete utterances [4]. By analyzing 790 student responses given to DeepTutor tutoring system [14], we have found that about 25% of the times even human needed additional information, such as the dialogue history in order to properly assess them [2].

As illustrated in Table 1, the student answers may vary greatly. For instance, answer A1 is elliptical. The *bug* in A2 is referring to the mosquito and *they* in A3 is referring

Rajendra Banjade and Vasile Rus "Assessing Student Response in Tutorial Dialogue Context using Probabilistic Soft Logic" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 258 - 263

Table 1: Some student answers to the given question asked at some point during interactions with DeepTutor.

Problem description: A car windshield collides with a mosquito, squashing it.
Tutor question: How do the amounts of the force exerted on the windshield by the mosquito and the force exerted on the mosquito by the windshield compare?
Reference answer: The force exerted by the windshield on the mosquito and the force exerted by the mosquito on the windshield are equal and opposite.
Student answers:
A1. <i>Equal</i>
A2. <i>The force of the bug hitting the window is much less than the force that the window exerts on the bug</i>
A3. <i>they are equal and opposite in direction</i>
A4. <i>equal and opposite</i>

to the amount of forces exerted to each other. Due to such variations in the answers, the semantic similarity methods alone can have issues in properly assessing those answers. For instance, the similarity between answer *A1* and the reference answer will be very low.

In this paper, we present Probabilistic Soft Logic (PSL; [3]) model for improving automatic assessment of open-ended answers in conversational ITS by augmenting the semantic similarity model with additional information, such as question difficulty and the knowledge level of the student. For instance, a high knowledge student answering many of the difficult questions correctly will probably answer the current question correctly. The proposed method allows us to model the complex interactions between the stochastic variables, such as student's knowledge level, question difficulty and the correctness of the student answer. In specific, the proposed PSL model which works on probabilistic reasoning framework allows us to concisely express our knowledge in First Order Predicate Logic (FOPL) rules and to provide the extent of our belief on such knowledge as weights. The inference is done over Probabilistic Graphical Model (PGM).

We evaluated our models on a dataset consisting of 790 responses collected during DeepTutor experiments and annotated for their correctness. The results show that augmenting the similarity model with question difficulty and knowledge level of the student improved the accuracy of our answer assessment model by about 8% when compared to results obtained using only the semantic similarity information.

2. RELATED WORK

Our work is more focused on assessing student responses in conversational tutoring systems. But most of the existing work has been performed on standard test taking environment (e.g., assignment checking). In this section, we briefly discuss approaches for constructed answer assessment where the student answers are short (one to just few lines) and the reference answers are available to compare with.

Martin *et al.* [9] proposed an assessment system OLAE using Bayesian nets. Latent Semantic Analysis (LSA; [8]) and

machine translation evaluation methods are also applied for answer grading. LSA method was also used in AutoTutor system [7].

Various researches show that the similarity based methods can be potentially used in the answer grading tasks [10, 15, 11]. In fact, a Semantic Evaluation (SemEval) shared task called Joint Student Response Analysis and 8th Recognizing Textual Entailment Challenge was organized in 2013 [5] to promote and streamline research in this area and almost all of the participating teams applied semantic similarity and textual entailment techniques.

Although various results show that the similarity based methods can be used in answer grading tasks, their implied assumption is that the text available are standard texts with noise filtered. Our work is focused on using naturally occurring texts from conversational tutoring systems where various linguistic phenomena are present, such as coreferences and ellipsis as discussed in Section 1. We also augment the semantic similarity based model using additional knowledge.

Furthermore, various datasets have been published over the years [12, 10, 5, 17]. But dataset from conversational systems with additional information (e.g., previous utterance, problem description, knowledge level of the student, question difficulty) are very limited. We annotated 790 student responses collected during an experiment with DeepTutor [14]. The dataset is made available for research purpose [2].

3. DATASET

We created the DT-Grade dataset [2] by extracting student answers from logged tutorial interactions between 36 junior level college students and the DeepTutor system [14]. During the interactions, each student solved 9 conceptual physics problems and the interactions were in the form of purely natural language dialogues, i.e., with no mathematical expressions and special symbols. We selected 790 answers for the annotation. We chose the more difficult ones (by observing responses from some students, the nature of the question, and so on) such that the similarity based models alone will have difficulty judging those answers.

Table 2: Summary of DT-Grade dataset.

Label	Count
All	790
Correct	319 (40.379%)
Correct but incomplete	292 (36.962%)
Incorrect	179 (22.658%)

Each instance contains the following information: (a) problem description (describes the scenario or context), (b) tutor question, (c) student answer in its natural form (i.e., without correcting spelling errors and grammatical errors), (d) list of reference answers for the question and has been assigned one of the following labels.

1. **Correct:** Answer is fully correct in the context. Extra information, if any, in the answer is not contradicting with the answer.
2. **Correct-but-incomplete:** Whatever the student provided is correct but something is missing, i.e. it is not complete. If the answer contains some incorrect part also, the answer is treated as incorrect.
3. **Incorrect:** Student answer is incorrect.

The dataset and further details about the collection and annotation of it can be found at [2].

4. PROBABILISTIC SOFT LOGIC MODELS

4.1 Background

Probabilistic Soft Logic (PSL; [3]) is an approach to combining knowledge in the form of first-order logic rules and probability in a single representation. It forms Probabilistic Graphical Models (PGM) which allow us to efficiently handle uncertainty and first-order logic allows us to compactly represent the knowledge. Furthermore, it allows us to jointly model the complex interactions among stochastic variables. For example, voting decision of friends has some influence on each other. Similarly, in answer assessment, a high knowledge student giving correct answers to the difficult questions will probably answer another difficult or easy question correctly and we can model such knowledge in a PSL model. On the other hand, typical machine learning algorithms assume that the data bear *i.i.d.* properties.

First-Order Knowledge Base. A first-order knowledge base (KB) is a set of formulas in first order logic [6]. Formulas are constructed using symbols: *constants*, *variables*, *predicates*, and *functions*. Constant represents an object (e.g., John). Functions represent mappings from tuples of objects to objects (e.g., *FatherOf*). Predicate represents relations among objects (e.g., *Friends*) or attributes of objects (e.g., *Smokes*). The formulas are typically written in clausal form (also known as conjunctive normal form (CNF)). For example,

$$Friends(x, y) \wedge Friends(y, z) \rightarrow Friends(x, z)$$

PSL Program. A PSL program consists of rules along with relative weights associated with them and the data (or

observations). The weights in the following example rules are assigned quite arbitrarily but they can be learned from the data which we discuss later.

$$5.0 : Friends(x, y) \wedge Friends(y, z) \rightarrow Friends(x, z)$$

$$2.0 : Friends(x, y) \wedge Colleague(y, z) \rightarrow Friends(x, z)$$

The rules are grounded using observations, i.e., each variable in the rules is assigned to all possible values in the observed data. For example, if there are three people: Joe, Bob, and Lili, then a grounded rule would look like,

$$5.0 : Friends(Joe, Bob) \wedge Friends(Bob, Lili) \rightarrow Friends(Joe, Lili)$$

Predicates in PSL program can have truth values in the range of [0 1], i.e. they are soft. For example, if it is not sure about the friendship of Joe and Bob but there is some possibility, then this uncertainty can be defined as a truth value in the range of 0 to 1. This is different from Markov Logic Network (MLN) where the predicates can have truth values either true or false (i.e., constraints in MLN are harder than PSL).

Prior Knowledge. The prior knowledge can also be encoded as rules in the PSL program. In our hypothetical example, let's assume that people who are neither friends of friends nor friends of colleagues can still be friends but the chances are very low. This can be expressed in the PSL program as illustrated below. It should be noted that the weight to our prior is very low as our belief is that any two persons being friends to each other (given no additional information) is possible but very less likely.

$$0.0001 : Friends(x, z)$$

As mentioned, the weights to the rules can be learned from the data itself. We discuss on this later. Next, we discuss some of the variables, predicates and rules we used in our PSL program (or model).

4.2 Model

Variables. Our model has two variables (*s* and *a*) and by convention, the variables are represented by lower case letters.

s - Student id, *a* - Answer id (or just id) which uniquely identifies an instance in the dataset. It should be noted that the question belonging to *a* may be same as that of some other answer id *b* because the same set of problems were attempted by multiple students.

Predicates. Following are the predicates used in our model.

- *SimilarityHigh(a)* $\in \{0, 1\}$ - similarity of answer *a* with corresponding reference answer is high
- *SimilarityMedium(a)* $\in \{0, 1\}$ -similarity of answer *a* with corresponding reference answer is medium

- $SimilarityLow(a) \in \{0,1\}$ - similarity of answer a with corresponding reference answer is low
- $PriorKHigh(s) \in \{0,1\}$ - prior knowledge of the student s is high
- $PriorKMedium(s) \in \{0,1\}$ - prior knowledge of the student s is medium
- $PriorKLow(s) \in \{0,1\}$ - prior knowledge of the student s is low
- $QDifficultyHigh(a) \in [0,1]$ - question difficulty is high (fraction of students who answered the question corresponding to a incorrectly)
- $QDifficultyMedium(a) \in [0,1]$ - question difficulty is medium (fraction of students who answered the question corresponding to a correctly but incompletely)
- $QDifficultyLow(a) \in [0,1]$ - question difficulty is low (fraction of students who answered the question corresponding to a correctly)
- $AttemptedBySameStudent(a,b) \in \{0,1\}$ - whether a and b were attempted by the same student
- $Correct(a) \in [0,1]$ - the truth value of answer a being correct
- $CorrectButIncomplete(a) \in [0,1]$ - the truth value of answer a being correct but incomplete
- $Incorrect(a) \in [0,1]$ - the truth value of answer a being incorrect

We have created three predicates for semantic similarity and for prior knowledge to avoid biases in assigning weights in some rules. For example, if we have rules 3.0 : $Similarity(a) \rightarrow Correct(a)$ and 2.0 : $Similarity(a) \rightarrow Incorrect(a)$, then low value of similarity score will still favor the first rule.

Rules and Priors. We present few rules with quite arbitrary weights. We learn the weights for those rules from the data which we present later in this section. The priors (starting with negation symbol \sim) specify the possibilities of being false. It should be noted that the weights are relative to each other and do not have to sum up to 1.

2.0 : $SimilarityHigh(a) \wedge QdifficultyLow(a) \rightarrow Correct(a)$
 3.0 : $SimilarityLow(a) \rightarrow Incorrect(a)$

...

0.002 : $\sim Correct(a)$

0.004 : $\sim CorrectButIncomplete(a)$

0.003 : $\sim Incorrect(a)$

4.3 Data

We used DT-Grade dataset described in Table 2. It includes responses provided by 36 students and we also had pretest scores for them. The pretest was a multiple-choice test which consisted of 39 questions.

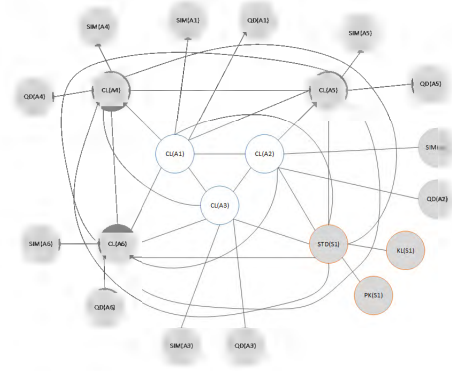


Figure 1: An illustration of a grounded probabilistic graphical network for a student. The shaded nodes are evidence nodes and non-shaded nodes are query nodes. CL - Correctness label, QD - Question difficulty, STD - Student, KL - Knowledge level, SIM - Similarity.

4.4 Grounding

During grounding phase, all the variables in the rules are substituted with possible values from the observations (i.e., data). Figure 1 illustrates an example of a grounded graphical network for a student's data but the graph can grow very large. For instance, the nodes corresponding to correctness labels of each answer are actually 3 (*Correct*, *CorrectButIncomplete*, and *Incorrect*) but in the graph they are represented by a single node *CL*. Similarly, the question difficulty *QD* has three values (high, medium, and low) and each one is actually represented by a separate node. Also, each student has attempted around 20 questions in average (counting those in the DT-Grade dataset only) which makes the graph bigger than what is shown in the figure. We discuss on the scale of the network in Weight Learning section.

The shaded nodes in the graph are observed nodes which we call **evidence**, whereas the light nodes are **query** nodes. During inference, the truth values of the query nodes are predicted jointly based on the evidence.

The similarity between student answer and the corresponding reference answer was calculated using optimal word alignment based method which has performed very well in general. We used the methods implemented in SEMILAR library [16]. We then grouped the similarity scores into high (score > 0.5), medium ($0.5 \geq \text{score} > 0.35$), and low (≤ 0.35) using empirically chosen threshold values. Similarly, we grouped the prior knowledge of the students into high (> 0.8), medium ($0.8 \geq \text{score} > 0.5$), and low (≤ 0.5) based on their pretest scores.

We calculated the question difficulty (high, medium, and low) as discussed in Section 4.2 (predicate definitions related to question difficulty). However, for question difficulty we have used soft values. In specific, each question has soft value (in $[0,1]$) for each of the difficulty levels: high, medium, and low. But for the difficult question, for example, the truth value of the predicate $QDifficultyHigh(a)$ will have higher value than the truth value of the predicates corresponding to other difficulty levels (medium, and low).

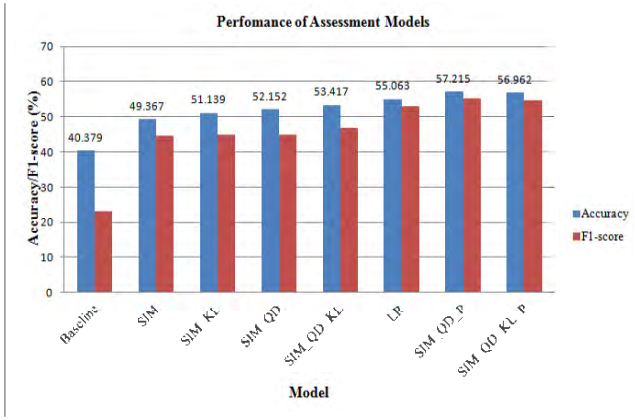


Figure 2: Results of different Probabilistic Soft Logic models on DT-Grade dataset. The models ending with *P* used the prior information learned using Logistic Regression (LR) models.

4.5 Weight Learning for PSL Rules

Laying out the internal details, the PSL rules' weight learning process is similar to typical supervised model learning process. We provide the ground truth (human annotated correctness label of the given answer) for the query predicate (which corresponds to a node in the grounded network graph). For each answer, there will be three query predicates one for each of the three labels. Since the labels are mutually exclusive, only one of them is set to 1.0 while other two will be set to 0.0. As we discussed earlier, the grounded network can become very large depending on the set of rules and the size of the dataset used to ground the rules.

Also, the same node cannot be a query node as well as evidence node at the same time. Therefore, we have replicated each student's data several times and renamed their ids such that we can make each answer in the original set (though renamed) a query at one time while using it as an evidence when other nodes are query nodes. This is important for us in both training (i.e., learning rules' weights) and evaluation phase because the dataset we used is comparatively small. For instance, if we make one answer for each student a query and keep others as evidence, then we will have only 36 records for the weight learning as well as for the evaluation. But making each node a query node at least once, we have the full dataset which is several times bigger than the aforementioned size and we can also evaluate our model using the full dataset (for example, by using leave-one-out approach). In another words, this process allows us to utilize the full set of data.

Just to get a sense of the scale of the graph, we assume that each student's data is replicated 5 times. Then the size of the graph (by taking the dominant term only) will be $(5 * 790) * (5 * 790) \sim 15$ million. Weight learning in such a huge probabilistic graphical model is impossible at least in our experimental settings. Therefore, we have pruned some rules that rapidly increase the size of the graph (e.g. the rules of the type: if answer to *a* is correct, then answer to *b* is also correct for the given student) and the resulting graph had about 200,000 nodes, on which we have managed

to learn the weights for the rules.

For those rules which rapidly increase the size of the network with increasing size of the data, we have learned the weights for each student and estimated the weights of the rules using weights learned at student level which is the sub-optimal solution. For each student, the average size graph had only few thousand nodes.

5. RESULTS

Including semantic similarity and additional information, we built several PSL models. For the experiments, we used the PSL tool¹ developed at University of Maryland, College Park. The tool uses Hinge-Loss Markov Random Fields (HL-MRFs) for inference and weight learning [1]. We set the number of iterations to be performed by the optimizer to 50,000.

By assuming that the performance of a student is independent of others, we refactored the graph into subgraphs one for each student and took the leave-one-student-out approach for PSL rules' weight learning and evaluation. As discussed in Section 4.5, we learned the weights for the rules from 35 students at a time (except for few rules for which weights were estimated using weights learned student-wise) and applied to the leave out student. Performing inference in such smaller graphs is computationally very efficient (takes few seconds for each student when run in a normal workstation). Also, the question difficulty was calculated based on training data only, i.e. using 35 students' data at a time.

Once inference is complete, i.e. the truth values for *Correct*, *CorrectButIncomplete*, and *Incorrect* predicates are assigned for each query answer. We then chose the correctness label corresponding to the highest truth value among those three. It should be noted that the truth value for each of them was in the interval $[0, 1]$ but their sum does not have to be 1.0. Then, we calculated the accuracy and F1 scores. The results of our various models are presented in Figure 2.

The baseline system is the majority class classifier, i.e. which labels each answer as correct. The accuracy of this baseline model was 40.379% which is equivalent to the percentage of correct answers in the dataset. *SIM* model used the similarity information only. It obtained 9% improvement over the baseline. As mentioned earlier, the DT-Grade dataset was developed by selecting the difficult cases, particularly difficult to judge by only comparing the student answer with the reference answer. Therefore, we consider 9% improvement in accuracy over baseline results as a notable improvement.

We then augmented the model using knowledge level (*KL*) of the student and question difficulty (*QD*). The *KL* includes the prior knowledge of the student which was assessed using multiple choice questions. The results were improved after adding question difficulty and knowledge level separately. Furthermore, when combined together our model achieved 53.417% accuracy which is above 4% improvement over results obtained using similarity information only.

In an another experiment, we used the priors learned using

¹<http://psl.linqs.org/>

Logistic Regression (LR). In specific, we obtained the probabilities (precisions) of any answer being *Correct*, *CorrectButIncomplete*, and *Incorrect* based on correctness predictions made by LR model when the given set of rules were used as features and used them in our PSL models as priors (model names ending *_P*). Since the priors in PSL models needed to be in negated form, we deducted each probabilities learned from LR from 1.0 and used the resulting values as priors. This has improved the results by about 5% in *SIM_QD* model and about 3% in *SIM_QD_KL* model. The results are above 7% when compared to *SIM* model. These results are also better when compared to the results of LR model itself. This shows that the LR model which is very different from PSL can complement the PSL model.

We learned priors using LR model only for *SIM_QD*. The *SIM_KL* included pretest scores as well as rules of the type: if answer to *a* is correct, then answer to *b* is also correct for the given student. Such rules that capture the relational dependencies are not easily modeled in Logistic Regression. Furthermore, the results of *SIM_QD_KL_P* is slightly less than *SIM_QD_P*. It seems that the concordance between the weights of the PSL rules and the priors learned separately may not be perfect in some cases.

6. CONCLUSION

We presented joint learning models using Probabilistic Soft Logic (PSL) for improving the assessment of open-ended student responses in conversational tutoring systems where the student responses can vary a lot. Specifically, our models augmented semantic similarity information with non-linguistic knowledge (student's knowledge level and question difficulty) and improved the accuracy of the assessment model when evaluated with DT-Grade dataset. The accuracy of our model using informed priors was up to 57.215%, which is more than 7% improvement over the results of semantic similarity based models. In the future, we intend to add additional information in the model and improve on PSL rules' weight learning by clustering students' data.

7. ACKNOWLEDGMENTS

This work was partially supported by The University of Memphis, the National Science Foundation (awards CISE-IIS-1822816 and CISE-ACI-1443068), and a contract from the Advanced Distributed Learning Initiative of the United States Department of Defense.

8. REFERENCES

- [1] S. H. Bach, M. Broecheler, B. Huang, and L. Getoor. Hinge-loss markov random fields and probabilistic soft logic. *arXiv preprint arXiv:1505.04406*, 2015.
- [2] R. Banjade, N. Maharjan, N. B. Niraula, D. Gautam, B. Samei, and V. Rus. Evaluation dataset (dt-grade) and word weighting approach towards constructed short answers assessment in tutorial dialogue context. 2016.
- [3] M. Brocheler, L. Mihalkova, and L. Getoor. Probabilistic similarity logic. *arXiv preprint arXiv:1203.3469*, 2012.
- [4] J. G. Carbonell. Discourse pragmatics and ellipsis resolution in task-oriented natural language interfaces. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, pages 164–168. Association for Computational Linguistics, 1983.
- [5] M. O. Dzikovska, R. D. Nielsen, C. Brew, C. Leacock, D. Giampiccolo, L. Bentivogli, P. Clark, I. Dagan, and H. T. Dang. Semeval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge. Technical report, DTIC Document, 2013.
- [6] M. R. Genesereth and N. J. Nilsson. Logical foundations of artificial. *Intelligence. Morgan Kaufmann*, 58, 1987.
- [7] A. C. Graesser, P. Wiemer-Hastings, K. Wiemer-Hastings, D. Harter, T. R. G. Tutoring Research Group, and N. Person. Using latent semantic analysis to evaluate the contributions of students in autotutor. *Interactive learning environments*, 8(2):129–147, 2000.
- [8] T. K. Landauer. Automatic essay assessment. *Assessment in education: Principles, policy & practice*, 10(3):295–308, 2003.
- [9] J. Martin and K. VanLehn. Student assessment using bayesian nets. *International Journal of Human-Computer Studies*, 42(6):575–591, 1995.
- [10] M. Mohler and R. Mihalcea. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 567–575. Association for Computational Linguistics, 2009.
- [11] N. Murrugarra, S. Lu, and M. Li. Automatic grading student answers. 2013.
- [12] R. D. Nielsen, W. Ward, J. H. Martin, and M. Palmer. Annotating students' understanding of science concepts. In *LREC*, 2008.
- [13] N. B. Niraula, V. Rus, R. Banjade, D. Stefanescu, W. Baggett, and B. Morgan. The dare corpus: A resource for anaphora resolution in dialogue based intelligent tutoring systems. In *LREC*, pages 3199–3203, 2014.
- [14] V. Rus, S. D'Mello, X. Hu, and A. Graesser. Recent advances in conversational intelligent tutoring systems. *AI magazine*, 34(3):42–54, 2013.
- [15] V. Rus and M. Lintean. A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 157–162. Association for Computational Linguistics, 2012.
- [16] V. Rus, M. C. Lintean, R. Banjade, N. B. Niraula, and D. Stefanescu. Semilar: The semantic similarity toolkit. In *ACL (Conference System Demonstrations)*, pages 163–168. Association for Computational Linguistics, 2013.
- [17] J. Z. Sukkarieh and E. Bolge. Building a textual entailment suite for the evaluation of automatic content scoring technologies. In *LREC*. Citeseer, 2010.
- [18] K. VanLehn, A. C. Graesser, G. T. Jackson, P. Jordan, A. Olney, and C. P. Rosé. When are tutorial dialogues more effective than reading? *Cognitive science*, 31(1):3–62, 2007.

Application of Hidden Markov Models to quantify the impact of enrollment patterns on student performance

Shahab Boumi
University of Central Florida
4000 Central Florida Blvd
Orlando, United States
sh.boumi@knights.ucf.edu

Adan Vela
University of Central Florida
4000 Central Florida Blvd
Orlando, United States
adan.vela@ucf.edu

ABSTRACT

Simplified categorizations have often led to college students being labeled as full-time or part-time students. However, at many universities student enrollment patterns can be much more complicated, as it is not uncommon for students to alternate between full-time and part-time enrollment each semester based on finances, scheduling, or family needs. While prior research has established that full-time students maintain better outcomes than their part-time counterparts, little study has examined the impact of mixed enrollment patterns on academic outcomes. In this paper, we apply a Hidden Markov Model to identify students' enrollment strategies according to three different categories: part-time, full-time, and mixed enrollment. According to the enrollment classification we investigate and compare the academic performance outcomes of each group. Analysis of data collected from the University of Central Florida from 2008 to 2017 indicates that mixed enrollment students are closer in performance to full-time students, than part-time students. More importantly, during their part-time semesters, mixed-enrollment students significantly outperform part-time students. Such a finding suggests that increased engagement through the occasional full-time enrollment leads to better overall outcomes.

Keywords

Hidden Markov model, student enrollment mode, academic outcomes

1. INTRODUCTION

In practice, either through choice or necessity [10, 15, 5, 11, 13], students engage in a variety of enrollment patterns over their academic career that includes full-time and part-time enrollment, or halting [13]. Based on a survey conducted at 253 academic institutions, only 18% of students maintain full-time status during all semesters they are enrolled, while 29% of students maintain part-time enrollment over their whole academic career. Meanwhile, the majority of

students, 59%, change their enrollment status between part-time and full-time at least once during their studies [1].

To date, part-time enrollment status has been indicated as risk-factor to student success. Feldman [8] shows that on average, at the end of the first academic year, full-time college students have higher retention rates and GPAs when compared to the part-time students. In another study, Pelkey [14] analyzed how race, age, enrollment status, GPA and financial aid can impact a student's persistence. Their analysis indicated that GPA and enrollment status, have the highest impact on persistence at college. Not only is enrollment status a factor but so is course-load, as demonstrated in [6], students with more credits during their first semester are more likely to complete their credits and degrees.

Despite it's perceived importance to student success there is no clear definition of what it means to be a *part-time* student or *full-time* student outside the ephemeral academic label. Given that the majority of students alternate between both enrollment statuses, it appears to be overly simplistic to group students together for analysis based on their enrollment status during a single semester; there is likely value in understanding more complex enrollment dynamics, and it's potential value in understanding student outcome. This assertion is supported by a 2015 nation-wide study indicating that student success can be found through mixed enrollment strategies [16] – the authors report that non-first-time-in-college students that attend college utilizing a combination of part-time and full-time enrollment are less likely to drop out and more likely to complete degrees when compared to full-time students.

In this study, we seek to find a more comprehensive means of identifying and clustering students with regards to their enrollment strategy (e.g. part-time, full-time, etc). Unlike a single-period model in which the students' strategy is equivalent to the observed student status (part-time or full-time), we make use of a multi-period dynamic approach using the Hidden Markov Models. Through application of the model we are able to provide a richer understanding of enrollment strategies, by extending our traditional notions to include not only full-time and part-time enrollment strategies, but also a mixed enrollment strategy. Students who use a mixed enrollment strategy regularly alternate between full-time and part-time status. After categorizing students into three groups of full-time, part-time and mixed enrollment strategy, we examine the student outcomes such as

Shahab Boumi and Adan Vela "Application of Hidden Markov Models to quantify the impact of enrollment patterns on student performance" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 264 - 269

Stu. #	Enroll. Status	Enroll. Strategy
1	F,P,F,F,F,F	F,F,F,F,F,F
2	F,P,F,P,F,P	M,M,M,M,M,M
3	P,F,P,P,F,P,P	P,P,P,P,P,P
4	P,F,F,P,P,P,P	M,M,M,P,P,P,P
Legend	FT=F, PT=P	FES=F, MES=M, PES=P

Table 1: Example enrollment status' over academic career and corresponding enrollment strategies

GPA and graduation rate associated with each strategy.

2. PROBLEM STATEMENT

We consider the problem of classifying students according to their enrollment strategy as opposed to their enrollment status during any given semester. For many students the distinction between enrollment strategy and actual enrollment is minor. At the University of Central Florida a students is considered as full time student in a given semester if he or she takes more than 12 credits in that semester. For approximately 35% of the student-body at the University of Central Florida, their enrollment status is consistently full-time throughout their academic career, meaning they employ a strategy of enrolling full-time. In contrast, the case for so-called *part-time* students it is not so clear. In any given semester, about 30% of enrollments are part-time, and yet only 7% of students consistently enroll part-time over their academic career. Enrolling part-time in any given semester is not equivalent to the strategy of consistently enrolling part-time. It follows that just because a student enrolls in a single semester part-time, that does not mean they bear similarity to student's who consistently enroll part-time.

The goal of this paper is to recognize and report the distinction between a student's enrollment strategy and enrollment status, and to find a more meaningful way to classify students over their academic career. More specifically, this paper develops a model that takes as its input a sequence of enrollment statuses and returns a sequence of estimated strategies applied over the same time-frame. In recognition that students apply a greater diversity of strategies then just a full-time enrollment strategy (FES) or part-time enrollment strategy (PES), we introduce the notion of a mixed enrollment strategy (MES). For a mixed enrollment strategy, students alternate between part-time and full-time enrollments. Table 1 provides examples of the enrollment status of four different students over their academic career along with the corresponding enrollment strategies. For example, enrollment strategies for student number 1 through number 3 are FES, MES, and PES respectively.

3. METHODOLOGY

In this study, we generate and apply a Hidden Markov Model (HMM) to identify students' enrollment strategy, and to characterize the impact of enrollment strategy on student outcomes. The use of HMM is not new to educational data mining and modeling. Previously it have been used to investigate students' sequential behaviors, decision-making, and performance [2, 3, 4, 9, 12, 7]. As an example Falakmasir et al. [7], classified students into low-performing and high-performing groups and applied and trained two hidden Markov models for each group separately. For each HMM,

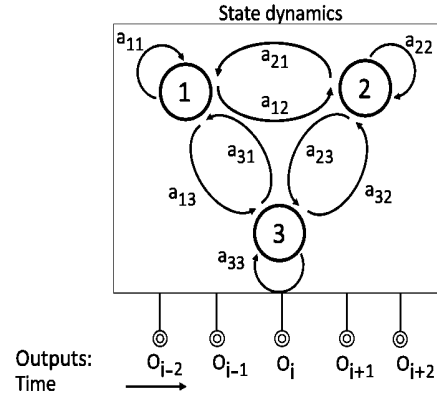


Figure 1: Representation of a simple Hidden Markov Model

they used forward algorithms to compute log-likelihoods for the observation sequences. They continued by applying a linear regression model to explain the difference between the computed log-likelihoods so as to predict post-test scores for the low-performing and high-performing students. Other papers have used HMMs in order to model sequential student behavior. Beal et al [4] colleagues modeled high school students' actions and behaviors using HMMs. By estimating HMM parameters with the Baum-Welch algorithm for each student, the authors clustered the students based on the individual transition matrices to assess differences in behavior and achievement of different clusters.

As depicted in Figure 1, similar to ordinary Markov Models, a HMM represents the dynamics of a system as it moves between operating states or modes (e.g. Modes 1, 2, and 3 in the figure). When operating within a state or mode, the system generates state-related output O_i at each time-step. Unlike Markov Models, in the case of the HMM problem the states are not always directly observable, and as such they can only be estimated by observing a sequence of outputs. For the problem under consideration here, the hidden state corresponds to the enrollment strategy of a student (e.g. full-time enrollment strategy, mixed enrollment strategy, part-time enrollment strategy), and the observations refer to the actualized enrollment in any given semester in the student academic history.

To give a formal definition of Hidden Markov models, we must begin with the following notations: $Q = \{q_1, q_2, \dots, q_N\}$ represents the set of N possible states in the system; $A = [a_{i,j}] \in \mathcal{R}^{N \times N}$ is a transition matrix, where each $a_{i,j}$ denotes the probability of transitioning from state i to j at any given time-step; $O = o_1, o_2, \dots, o_T$ represents a sequence of observations of length T , each drawn from the set of M possible observations $V = \{v_1, v_2, \dots, v_M\}$; and π represents the distribution of the initial state the system begins in. When a system is operating in a specific state q_i , the output o_t at any given time t is generated according to a unique probability distribution denoted as $B = b_i(o_t)$, the emission probability.

In order to generate a HMM to represent student enrollment strategies, we must learn the optimal model parameters $\lambda = (A, B, \pi)$ that reproduce known observations. The

process of learning λ is based on the Baum-Welch algorithm, which is an iterative process that requires calculating the likelihood of any sequence of observations given λ , and decoding relationships between observations and hidden variables. As the model is iteratively updated, the likelihood calculations and the decoding is updated.

4. STUDENT DATA RECORDS

The study presented in this paper makes use of processed undergraduate student records collected from the University of Central Florida, a large public university in the southeast United States, between the years of 2008 to 2017. The total data-set amounts to approximately 170000 records. The data set contains a wide variety of information about students at UCF, including but not limited to: (1) demographic information, (2) admission information for students who have been admitted and enrolled, (3) degrees awarded (for bachelor level), (4) courses taken by student at UCF, and (5) family income. Some of the demographic information along with the fraction of students who enroll as full-time and part-time, and admission type (FTIC and transfer) are provided in Tables 2 through 5.

Table 2: Students gender distribution at UCF over 10 years

	Females	Males
Percentage	56%	44%

Table 3: Students ethnicity distribution at UCF over 10 years

	White	Hispanic	African-Am.	Other ¹
Percentage	55%	24%	11%	10%

Table 4: Students admission type distribution at UCF over 10 years

	First-Time-in-College	Transfer
Percentage	41%	59%

The processed student data includes a unique identifier, along with the student's observed academic load for semester they enrolled. Synthetic examples are shown in Table 1. For each student their enrollment sequence is ordered from their first observed enrollment to their last observed enrollment without making note of the semester or year. The data set includes both partial, halted, and graduated enrollment sequences within the indicated 10 years date-range. For the purposes of this study we restrict the problem to enrollment during Fall and Spring semesters, as such information regarding Summer enrollment is excluded when constructing the HMM. It is worth noting that the data-set includes both first-time-in-college students and transfer students.

¹The other category includes American-Indian, Asian, Native Hawaiian, and Multi-racial ethnicity

Table 5: Enrollment type distribution for different semesters at UCF over 10 years

Semester	Full-time	Part-time
Fall	73%	27%
Spring	71%	29%
Summer	10%	90%

5. APPLYING HMM TO STUDENT DATA

In applying the HMM model to our problem, we begin by identifying the set of hidden states corresponding to three different enrollment strategies: full-time enrollment strategy (FES), part-time enrollment strategy (PES), and mixed enrollment strategy (MES). The probability a student changes his or her enrollment strategy from one semester to the next is represented using a probability transition matrix A . While the probability of observing an enrollment status while using a specific enrollment strategy is given by the emission matrix B . Finally, π is the probability distribution over the students enrollment strategy during their first enrolled semester.

Beginning with an initial guess for A , B , and π , the Baum-Welch algorithm is applied to estimate the true model parameter set (λ). Converging after 20 iterations, the following values for A , B , and π are generated:

$$A = \begin{bmatrix} 0.898 & 0.05 & 0.052 \\ 0.168 & 0.74 & 0.092 \\ 0.007 & 0.12 & 0.873 \end{bmatrix} \quad (1)$$

$$B = \begin{bmatrix} 0.974 & 0.026 \\ 0.611 & 0.389 \\ 0.061 & 0.939 \end{bmatrix} \quad (2)$$

$$\pi = [0.718 \quad 0.113 \quad 0.169] \quad (3)$$

For any two subsequent semesters t and $t + 1$, the rows in the transition matrix A correspond to states FES, MES and PES at semester t , while the columns correspond to states FES, MES and PES at semester $t + 1$. Based on the estimated transition matrix A , most of the students maintain their enrollment strategy with high probabilities from one semester to the next. Reading the diagonal of the matrix, with .898 probability a student employing a FES will continue employing a FES, similarly .74 for PES and .873 for MES. This indicates that most students maintain a static enrollment strategy, even if it is a mixed one.

For emission matrix B , each rows correspond to the probability of full-time and part-time enrollment status in a semester for a given enrollment strategy. Result indicate that students employing FES register full-time with probability 0.974 and as part-time with probability 0.026. While students employing a PES only register full-time with probability 0.061 versus part-time at 0.939. Most interesting are students with MES, as their full-time and part-time enrollment is split between 0.611 and 0.389. The, initial probabilities matrix π indicates that most of the undergraduate students start their first semester with full-time enrollment strategy (with probability 0.718). Moreover, the probability of being at PES

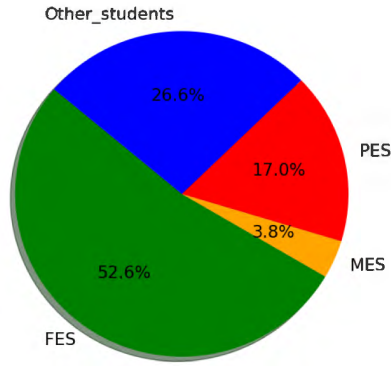


Figure 2: Distribution students' enrollment strategy

and MES at the first semester are 0.169 and 0.113 respectively.

6. ANALYSIS

After estimating model parameters, the next step is to find the strategies (hidden states) for each student in the data set at each semester with the Viterbi algorithm. Based on the estimated hidden states, students are classified into four groups, three of which corresponds to the students who maintain a consistent strategy of FES, PES or MES during their education. The last group corresponds to students who employee a combination of FES, MES and PES over their academic career. Figure 2 shows how students are distributed among these four groups.

Based on Figure 2, most of the students maintain their enrollment strategy during their educational career (Sum of green, red and yellow slices, approx. 73.4%). The most prevalent consistent enrollment strategy is FES, followed by PES and MES groups. For those students that change strategies at some point in the academic career, 73%, change from FES to PES, and 15% move from PES to MES. For virtually all cases of FES to PES, the majority of students adjust their enrollment strategy during their last two semesters. Anecdotally, it appears this shift is due to course scheduling inefficiencies and early entrance into the work place through co-op placements.

Furthermore, Table 6 represents percentage of male and female students for different enrollment strategies. The percentage of female students in FES, MES, and PES groups are 55%, 54%, and 55% respectively, which emphasizes that students enrollment strategy is independent of students gender. Table 7 indicating how students with different ethnicity are distributed among the three enrollment strategy groups. As the table shows, the ratio of students with white and Hispanic ethnicity in FES group are different to MES and PES groups. Hypothesis *t*-tests are conducted to assess statistical significance of these differences. For FES and PES groups, the *p*-value is close to 0, implying the difference in the ratios are statistically considerable. However, other complicating factors have not been considered.

Clustering of the students based on enrollment strategy (FES, PES, MES), a number of descriptive statistics are calculated.

Table 6: Female and male ratios for students with different enrollment strategies

Strategy	Female	Male	Number of students
FES	55%	45%	74571
MES	54%	46%	5321
PES	55%	45%	20466

Table 7: Ethnicity ratios for students with different enrollment strategies

Strategy	White	Hispanic	African-Am	Other
FES	56%	22%	12%	10%
MES	50%	27%	13%	10%
PES	50%	27%	13%	10%

They include average cumulative GPA, family income, 6-year graduation rate, and halting. The average GPA for each strategy cluster is shown in Figure 3. Results show that the FES group has the highest average GPA. The lowest GPA corresponds to the PES group, while the MES group's GPA lies in between.

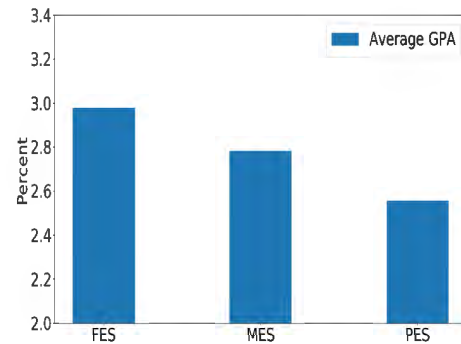


Figure 3: Average GPA for different enrollment strategies

To assess if the average GPA for each group are statistically different from the other groups, statistical hypothesis *t*-tests are conducted. The result shows that the *p*-values for all the hypothesis tests are nearly to 0, indicating that the average GPA for each group is statistically different from others. The results are summarized in Table 8.

Furthermore, inside each strategy cluster, the average GPA during full-time and part-time semesters are calculated. As indicated in Figure 4, for the FES group the average GPA for full-time and part-time semesters are 3.1 and 2.8. This indicates that students employing a full-time enrollment strategy, tend not to perform as well when registering part-time. The same conclusion is observed for students in the PES group, that is, student utilizing a part-time enrollment strategy perform better when they enroll full-time². Of interest

²For both FES and PES, comparison of GPAs between full-time and part-time semesters through difference of means statistical tests rejects the null hypothesis that the means are equal, $P = .001 < .05$

Table 8: Results for the GPA hypothesis tests

Pair of groups	P-value
FES and PES	0
FES and MES	$6.5e^{-84}$
MES and PES	$1.82e^{-86}$

Table 9: Results for the family income hypothesis tests

Pair of groups	P-value
FES and PES	0.7463
FES and MES	0.6518
MES and PES	0.9603

however, is that for students employing a mixed-enrollment strategy hypothesis tests indicate there is no statistical difference in means between the average GPAs of full-time and part-time semesters; in other words, the semester enrollment status for MES students does not significantly impact their GPAs. While the GPA reductions observed for students employing a full-time enrollment strategy and part-time enrollment strategy appear reasonable, the lack of GPA drop for mix enrollment strategy students is somewhat surprising, it suggests potential value in encouraging part-time students to occasionally enroll full-time.

Next, the impact of the family financial status on student enrollment strategy in each group is compared. As shown in Figure 5 the annual family income for students in all three groups of FES, MES, and PES are close to \$75000. This implies that at UCF, students enrollment strategy is independent to the family income. Kolmogorov-Smirnov (K-S) test is applied in order to assess if there is statistically significant difference in family income distribution for students with different enrollment strategy. As shown in Table 9, the hypothesis test results p-values greater than 0.05 for all three group pairs of indicating no significant difference in annual family income between students with different enrollment strategies.

The next criteria for comparing student performance be-

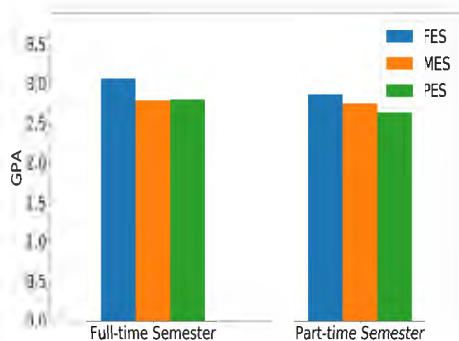


Figure 4: Average GPA for full-time and part-time semesters based on employed strategy

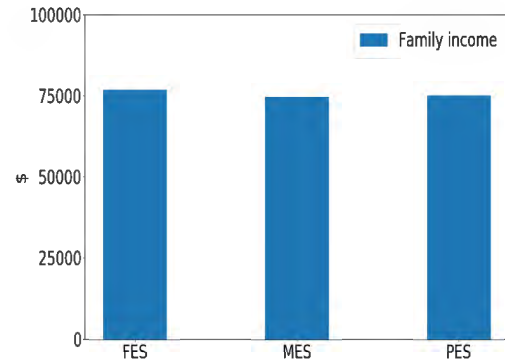


Figure 5: Annually family income for different enrollment strategies

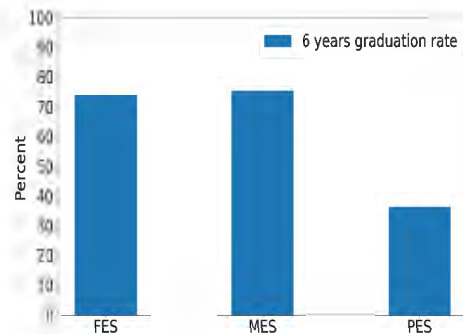


Figure 6: 6 year graduation rate for students utilizing different enrollment strategies

tween the three different groups is the 6-year graduation rate, summary statistics provided in Figure 6. As the plot shows, PES group has a lower graduation rate (37.3%) when compared to MES and FES groups (with similar graduation rates of approximately 75% and 74%). While the performance difference between PES and FES follows prior studies, it is interesting to note that employing a mix enrollment strategy does not appear to hinder graduation rates³.

7. CONCLUSION

The long-term vision of this research is to help identifying strategies that engender student success. Towards that end, this paper examined different enrollment strategies students apply over their academic career. Through application of Hidden Markov Models on a large student data set, we noted three dominant consistent strategies: full-time enrollment strategy, part-time enrollment strategy, and mixed enrollment strategy. The resulting HMM and its application leads to the conclusion that most of the students have a full-time enrollment strategy. When assessing different features of the three different enrollment strategies, we observe that the average GPA for FES students is the highest, followed by MES and PES students. While graduation rates indicate that students employing the PES are more at risk of

³Difference of proportions fails to find a difference between FES and MES graduation rates, $P=.112 > .05$

not graduating college. Also, financial analysis shows that there is no statistically significant difference between family income distributions for students with different enrollment strategies.

The major contributions of this research is twofold. Firstly, we provide a powerful tool for identifying students enrollment strategy as FES, PES or MES, based on their historical enrollment status. Secondly, our multi-aspect assessments on each group of students, emphasizes the vulnerability of the PES group, while encouraging university to policy-makers identify such students early during their studies and help them shift towards a mixed enrollment strategy by providing them with financial, educational, and social support.

8. REFERENCES

- [1] Center for community college student engagement. even one semester: Full-time enrollment and student success. the university of texas at austin, college of education, department of educational administration, program in higher education leadership. 2017.
- [2] S. Abdi, H. Khosravi, and S. Sadiq. Predicting student performance: The case of combining knowledge tracing and collaborative filtering.
- [3] C. Beal, S. Mitra, and P. Cohen. Modeling learning patterns of students with a tutoring system using hidden markov model, in proceedings of the 13th international conference on artificial intelligence in education (aied), r. luckin et al.(eds), marina del rey, july 2007.
- [4] K. E. Boyer, R. Phillips, A. Ingram, E. Y. Ha, M. Wallis, M. Vouk, and J. Lester. Investigating the relationship between dialogue structure and tutoring effectiveness: a hidden markov modeling approach. *International Journal of Artificial Intelligence in Education*, 21(1-2):65–81, 2011.
- [5] A. F. Cabrera, K. R. Burkum, S. M. La Nasa, and E. W. Bibo. Pathways to a four-year degree. Technical Report ED 482 160, 2003.
- [6] R. Darolia. Working (and studying) day and night: Heterogeneous effects of working on the academic performance of full-time and part-time students. *Economics of Education Review*, 38:38–50, 2014.
- [7] M. H. Falakmasir, J. P. González-Brenes, G. J. Gordon, and K. E. DiCerbo. A data-driven approach for inferring student proficiency from game activity logs. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 341–349. ACM, 2016.
- [8] M. J. Feldman. Factors associated with one-year retention in a community college. *Research in Higher Education*, 34(4):503–512, 1993.
- [9] D. Halpern, S. Tubridy, H. Y. Wang, C. Gasser, P. O. Popp, L. Davachi, and T. M. Gureckis. Knowledge tracing using the brain. In *Proceedings of the 11th International Conference on Educational Data Mining, EDM*, 2018.
- [10] J. C. Hearn. Attendance at higher-cost colleges: Ascribed, socioeconomic, and academic influences on student enrollment patterns. *Economics of education review*, 7(1):65–76, 1988.
- [11] J. C. Hearn. Emerging variations in postsecondary attendance patterns: An investigation of part-time, delayed, and nondegree enrollment. *Research in Higher Education*, 33(6):657–687, 1992.
- [12] N. Hoernle, K. Gal, B. Grosz, P. Protopapas, and A. Rubin. Modeling the effects of students’ interactions with immersive simulations using markov switching systems. *Proceedings of Educational Data Mining*, 2018.
- [13] D. M. O’toole, L. S. Stratton, and J. N. Wetzel. A longitudinal analysis of the frequency of part-time enrollment and the persistence of students who enroll part time. *Research in Higher Education*, 44(5):519–537, 2003.
- [14] D. Pelkey. Factors supporting persistence of academically underprepared community college students. 2011.
- [15] S. F. Reardon, R. Baker, and D. Klasik. Race, income, and enrollment patterns in highly selective colleges, 1982-2004. Technical report, Center for Education Policy Analysis, Stanford University., 2012.
- [16] I. Track. National study of non-first-time students shows full-time enrollment may not be appropriate for all, 2015.

Design and evaluation of a semantic indicator for automatically supporting programming learning

Julien Broisin
Université Toulouse 3 Paul Sabatier, IRIT
118 route de Narbonne
31062 Toulouse, France
julien.broisin@irit.fr

Clément Hérouard
Ecole Normale Supérieure
Campus de Ker lann, Avenue Robert Schuman
35170 Bruz, France
clement.herouard@ens-rennes.fr

ABSTRACT

How to support students in programming learning has been a great research challenge in the last years. To address this challenge, prior works have mainly focused on proposing solutions based on syntactic analysis to provide students with personalized feedback about their grammatical programming errors and misconceptions. However, syntactic analysis falls short on informing learners how they solve the programming problem, even if one key learning outcome of programming relates to the development of an individual's ability to solve a problem. In this article, we introduce an indicator to analyze beginners' code based on semantic proximity. This indicator adapts an edit distance algorithm (i.e., the Levenshtein distance) to express the proximity of the students' code with the expected solution provided by the teacher, in order to express the learners' capacity to solve the given problem. To process our indicator, we applied machine learning techniques to a dataset from an introductory programming course with a sample of 166 students. The first results are encouraging. On the one hand, the semantic indicator can be used to automatically classify source codes as semantically correct or incorrect in 58% of the cases. On the other hand, the indicator is correlated with teachers' summative evaluations of students' codes. Even if further investigations must be conducted to improve the indicator's accuracy, the results of this study make it possible to use our approach as the foundations for future research in semantic-based intelligent and awareness programming systems.

Keywords

Programming learning, Semantic analysis, Educational data mining, Edit distance, Levenshtein algorithm

1. INTRODUCTION

Computer literacy is currently booming. In Europe, particularly in Germany and the United Kingdom, profound educational transformations have been initiated since 2016 to promote digital learning in schools and prepare learners

for the acquisition of 21st century skills, which include programming learning. In France, for example, an educational reform of High School curricula that will be operational next year offers a Digital and Computer Science option that includes more than 350 hours of programming learning. This interest in integration of programming learning skills early in the curriculum requires not only prepared teachers, but also technological solutions to support them and their students in their daily practices.

With this purpose, the Technology-Enhanced Learning research community has been interested in designing systems dedicated to support learning of programming, as evidenced by different efforts intended to analyze learners' behaviour [14, 12]. One of the most common approaches in prior works consists in designing systems that analyze learners' programming codes from a syntactical perspective. These systems make a syntactic evaluation of students' codes to detect grammatical errors and provide appropriate and constructive support to learners to avoid misconception errors [15]. However, delivering feedback about syntactical errors falls short on providing meaningful information about how students approach the programming problem. Yet, one of the key learning outcomes of programming relates to the development of an individual's ability to solve a problem [11]. One approach to achieve this goal is to design systems able to analyze source codes from a semantic perspective, i.e., able to show how the problem has been solved. Although there have been some initiatives approaching code evaluation from a semantic perspective [5], works on this line are still scarce and very few solutions have been proposed. Thus, more solutions based on semantic analysis are needed to better understand the potential of this approach in supporting learning of programming.

To advance on semantic analysis-based solutions, this article introduces the design of an indicator revealing the semantic proximity of two distinct source codes in order to express the correctness of a learner's code regarding a given problem, and tackles the following research questions:

Research question 1: How to design a semantic indicator revealing learner's ability to solve a problem?

Research question 2: Is the semantic quality of a learner's production correlated with his/her academic performance?

To answer the first question, we adopt an approach based on the comparison of abstract syntax trees. We adapt the edit distance established by Levenshtein, an algorithm that

Julien Broisin and Clément Hérouard "Design and evaluation of a semantic indicator for automatically supporting programming learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 270 - 275

has proven effective in comparing and correcting strings of characters. Then, we propose a machine learning method to determine some of the factors required to process the semantic proximity using source codes produced by 166 students. For the second research question, and with the objective of evaluating the precision of the indicator in relation to human perception, we conduct a set of statistical analysis with source codes gathered from an authentic learning context. In particular, we analyze the correlation between the value of our indicator automatically processed, and the teachers' scores manually assigned to students' source codes.

In the following section we review what current approaches dedicated to automatic source code analysis are currently available in the literature, and highlight a lack of semantic-driven proposals. Then, in Section 3, we present how the semantic indicator was designed; as stated above, the indicator stands on the Levenshtein distance adapted to source codes comparison, and depends on a set of parameters leading to its calculation. Section 4 then introduces the machine learning method we used to assign values to these different parameters, whereas Section 5 describes the dataset used to actually determine the values of the parameters. Section 6 gives the results we obtained, and evaluates the quality of the indicator at two different levels: its capacity to act as a semantic classifier, and its correlation with human perception. Finally, we discuss the results of this study as well as the main conclusions of the work.

2. APPROACHES FOR AUTOMATIC SOURCE CODE ANALYSIS

In the context of programming learning, automatic source code analysis is used to achieve different objectives such as improving feedback provided to learners [7, 9], predicting their performance [1, 16], or monitoring their activities [3].

These automatic analyses are performed by compiling the code [6], by searching for typical errors within a source code [9], or by running unit tests provided by teachers [13]. These various works guide learners in the production of syntactically correct programs, but they do not allow the source code to be evaluated at a semantic level: syntactic evaluation is not sufficient to reflect the relevance of a learner's production regarding a problem given by the teacher.

In the meta-review proposed by Ihantola et al. [8] who studied no less than 118 research articles in the field of educational data mining for programming, the word *semantics* is missing from the paper. Also, a search for the terms "semantic analysis programming" in Google Scholar and Web of Science returns a large number of results, but no scientific articles really deal with semantic analysis of code.

The works we have identified that are close to a semantic analysis are those proposed by Bey et al. [5]. In order to propose an automated assessment of learners' code in a Massive Open Online Course (MOOC), the authors propose to compare the control flow graph matching with the code produced by the learner, with a set of graphs stored in a database and manually assessed by experts [2]. If the learner's graph is recognized among the graphs of the database, then the matching score is returned to the learner; if it is not recognized, then the learner's production is manually assessed by

a human expert to enrich the database of graphs.

The lack of work addressing automatic analysis of source code from a semantic perspective can be explained by the fact that the semantics of a program can not be calculated. Our research try to go beyond this limitation and propose the design of an indicator that reveals the relevance of a source code regarding a given problem.

3. SPECIFICATION OF THE SEMANTIC INDICATOR

Our objective is to design an indicator able to evaluate the distance between two source codes from a semantic perspective and fulfilling the following requirements: (i) teachers should only produce a solution to the problems delivered to students, in contrast, for example, to methods based on unit tests which are time consuming; (ii) the distance should decrease as the student approaches the solution, so that an incomplete script can be evaluated even if it does not fully address the problem posed. Therefore, if the solution of a problem is provided by the teacher, then the distance between this source code and the learners' productions should give insights about their ability to solve the problem.

The field of natural language processing has for a long time studied problems related, for example, to spelling correction where semantics is naturally taken into account [4]. To design our semantic indicator, we studied algorithms capable of processing the edit distance between two sequences of characters, and built our proposal upon the Levenshtein algorithm which has been shown very useful for calculating the distance between two strings of characters.

3.1 Levenshtein distance

The edit distance between two strings of characters, or Levenshtein distance [10], is defined by a cost calculated from the minimum number of operations (i.e., insertion and deletion of a character, and substitution of one character by another) required to move from one string to another. In the case of character strings comparison, the costs associated with each of these operations are all set to 1, but the cost of the substitution of a character by itself which is 0. To apply this algorithm to source code, two challenges must be tackled: (i) the design of a formal representation of the source code in order to make it comparable at a high abstraction level, and (ii) the assignment of a specific cost to each elementary operation according to the importance of the modifications to be made to move from one source code to another. Indeed, substituting two instructions implementing the same functionality should not have the same cost than replacing an instruction with another one characterized by a very different functionality.

3.2 Formal representation of source code

Our proposal for building a formal representation of source code relies on two steps: transformation of the source code as an Abstract Syntax Tree (AST), and transformation of the AST into a string of elementary instructions.

3.2.1 Production of the abstract syntax tree

We have adopted abstract syntax trees to formally represent a source code as they do not represent nodes and branches

that do not affect the semantics of a program. We rely on a parser written in JavaScript that reads a script and returns the matching abstract syntax tree formatted as a JSON object; an example illustrating the conversion of a Bash source code to an AST is given in Figure 1.

With the formal representation proposed above, the measurement of our indicator corresponds to the edit distance between two abstract syntax trees. However, in order to adapt Levenshtein distance to our context, we need to convert the AST generated from the source code into a string of characters; from a terminology point of view, we name *tokens* the characters resulting from the AST transformation process so as not to confuse them with those contained in the source code. Thus, we carry out an in-depth exploration of the AST to create the string of tokens.

3.2.2 Production of the string of tokens

The Bash language, the subject of our study, proposes 17 control structures, each corresponding to a different token. A token can be of the type **Command**, of the type **Assignment**, or of the type **While**, **Do** or **Done** to indicate the beginning, body or end of a loop respectively. Note that the tokens **Command** keep the name of their command and arguments. The transformation of the AST represented in Figure 1 into a string of tokens is illustrated in Figure 2.

The Levenshtein distance between two strings of tokens is therefore characterized by a number of costs, or parameters, matching with the different elementary operations of insertion, deletion and substitution of each token. The 17 tokens of the Bash language lead to the definition of 308 costs: 17+17 costs to create and delete a token, 16x17 costs to substitute one token by another, 1 cost to substitute a token **Command** by another characterized by a different command name, and finally 1 cost to substitute a token **Command** by another characterized by identical names and different arguments. This number of parameters is too high to implement an effective machine learning method, as it decreases the density of tests and makes more difficult the search for "good" values of parameters.

3.3 Reduction of the number of parameters

A first operation to reduce the number of parameters to be trained consists in ignoring the tokens **Until**, **Subshell** and **Pipeline** because the matching control structures do not appear in the dataset used for this study (see Section 5). We also propose to symmetrize the problem by assigning equal costs to the insertion and deletion operations of a token; similarly, the substitution of the token *A* by the token *B* has a cost equal to the substitution of *B* by *A*. This symmetrization makes the distance between the scripts S_1 and S_2 equal to the distance between S_2 and S_1 , and reduces the number of parameters to 107.

Finally, we assume a stronger hypothesis by considering that only certain tokens can replace others. Thus, only the following groups of tokens can substitute each other: {**Command**, **Assignment**}, {**If**, **Case**}, {**Then**, **Else**, **CaseItem**}, and {**For**, **While**}. Indeed, it is possible that a script contains a **Case** where another script uses a **If**, but it is unlikely that a substitution of a **For** by a **If** appears frequently, these two instructions having very different objectives.

These simplifications considerably reduce the number of costs, since 23 parameters must now be calculated. Let us note θ the vector containing these 23 costs. Our indicator, noted $d(S, C, \theta)$, is then defined by the edit distance, under the 23 parameters θ , between the string of tokens representing the script S and the one matching with the script C . The next section introduces the machine learning criterion leading to the optimal values of θ .

4. MACHINE LEARNING METHOD

To train the correct values of the 23 costs of the θ parameter, we define the machine learning criterion $Score(\theta)$ to be minimized such that:

$$Score(\theta) = \frac{\sum_{(S,C) \in Correct} \sqrt{d(S, C, \theta)}}{\sum_{(S,C) \in Incorrect} \sqrt{d(S, C, \theta)}} \quad (1)$$

where *Correct* (resp. *Incorrect*) is the set of pairs composed of the correct (resp. incorrect) scripts of the learning dataset (see next section) and the associated corrections.

The *Score* function is low when the correct scripts are associated to short distances, and the incorrect scripts to long distances. We add the square roots of the distances to reduce the influence of high values.

We can notice a property of the function d :

$$\forall \lambda \in \mathbb{R}_+^*, d(S, C, \lambda \cdot \theta) = \lambda \cdot d(S, C, \theta) \quad (2)$$

Then it is obvious that $Score(\lambda \theta) = Score(\theta)$. This means that the *Score* function is constant on all rays from 0, and that exploring the different costs of the θ parameter, noted θ_i such as $0.01 < \theta_i \leq 1$, is sufficient to find the values of θ_i minimizing *Score*. The dataset built from an authentic learning situation and that was used to determine the θ_i is described in the next section.

5. DATASET FOR CALCULATING COST

The dataset was obtained in an IT department of Higher Education Institute of Technology (HEIT) during an introductory course on Bash programming attended by 166 first year students. Learners discover common commands such as **echo**, **ls**, **read** or **cat**, variable management, as well as some control structures of the Bash language (e.g., **for**, **while**, **if**, **case**). Students then put these concepts into practice during hands-on sessions where they produce Bash scripts to try to solve a series of problems.

The dataset includes the students' scripts produced during five practical sessions of one hour and a half. Each time a student saves the modifications of her Bash script, a copy of the script together with its timestamp and the identifier of the student is saved in a directory to enrich the dataset. Thus, our sample includes 19232 scripts. However, a number of nomenclature errors or file numbering made 18% of these scripts unusable: the dataset is composed of 15794 scripts.

This sample includes, for the same exercise, several scripts produced by the same student. However, machine learning methods are effective when they are based on independent data. We thus only consider, per exercise and per student, the last script produced by the learner because we assume

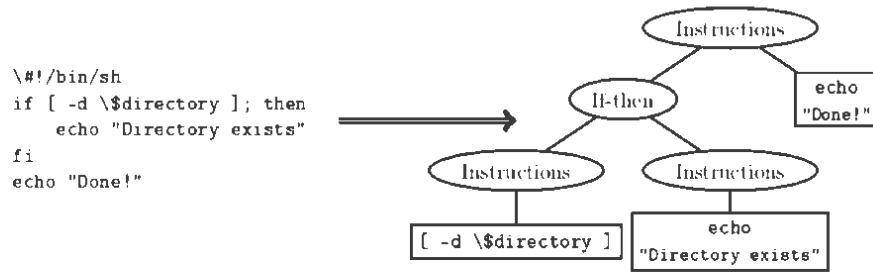


Figure 1: Converting a source code into an abstract syntax tree.

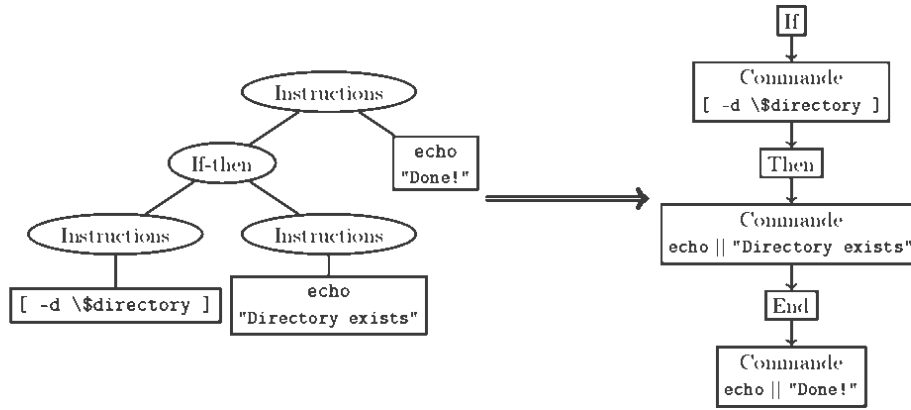


Figure 2: Converting an abstract syntax tree into a string of tokens.

that it is the most complete. This choice allows to obtain an independent dataset, but reduces the sample to 2540 scripts.

Finally, we want to be able to manually classify each script of the dataset into two distinct categories (i.e., *Correct* and *Incorrect*) expressing their semantic accuracy, in order to evaluate the accuracy of our indicator as a semantic classifier (see next section). However, some exercises suggested by the teachers of this course do not allow for a semantic analysis of students' productions. For example, some problems are related to the understanding of the execution of a script provided in the statement, while others have instructions that are too open to assess whether a script represents a solution to the problem. After this classification process, the final sample includes 733 scripts spread over 11 different exercises; 397 scripts are semantically correct, the others being semantically incorrect.

This dataset was randomly divided into a learning dataset whose objective is to identify the values of the θ parameter that minimize the machine learning criterion, and a test dataset that aims at evaluating the relevance of the results obtained during the machine learning phase. These two samples contain the same number of scripts for each exercise, and the proportion of semantically correct and incorrect scripts is preserved. In addition, let us note that a solution was provided by a teacher for each of the exercises.

6. RESULTS AND EVALUATION

6.1 Values of costs

The learning dataset was used to initialize the gradient descent minimization algorithm. The vector minimizing the

Score function, noted $\bar{\theta}$, admits values of 0.01 (i.e., the minimum value we have set) almost everywhere except for three parameters: (1) the creation or deletion of a token **Command**, whose cost is 1; (2) the creation or deletion of a token **If**, whose cost is 0.112; (3) the creation or deletion of a token **Function**, whose cost is 0.022.

At this stage, we have the parameter $\bar{\theta}$ which contains the values of the 23 costs that were obtained by our learning method. The objective of the tests conducted in the following section with the parameter $\bar{\theta}$ on the test dataset is twofold: to study the ability of our indicator to distinguish a semantically correct script from an incorrect script ; to study the correlation between our indicator and manual scores assigned to scripts by teachers.

6.2 Study of prediction

A first approach to assess the relevance of our indicator under the parameter $\bar{\theta}$ is to evaluate its ability to automatically classify a script as correct or incorrect from a semantics point of view. In a first step, we calculate the distance $d(S, C, \bar{\theta})$ between each script S of the test dataset and the correction C of the matching problem, in order to obtain a value of our indicator for each correct and incorrect script.

The next step consists in evaluating the prediction model resulting from the different values of the indicator. To do this, we use the ROC metric, which is widely used to observe the performance of a binary classifier. When the classifier calculates a metric m that is compared to a σ threshold to predict the class, the idea of the ROC curve is to vary σ from 1 to 0 and, for each σ value, plot the false positive rate on

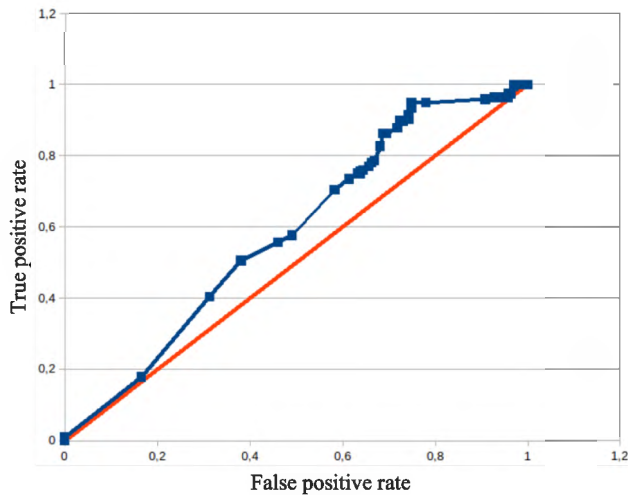


Figure 3: ROC curve of our classifier (in blue) and random bisector (in red).

the abscissa and the true positive rate on the ordinate. The area under the curve (AUC) then indicates the probability that the classifier will place a positive in front of a negative. A classifier that is never wrong has an AUC equal to 1, while a random classifier has an AUC equal to 0.5.

In our study, if $d(S, C, \bar{\theta}) < \sigma$, then the script is classified in the category *Correct*; otherwise it is classified in the category *Incorrect*. The ROC curve in Figure 3 illustrates the ratio of true positives (i.e., fraction of correct scripts that are actually detected as correct by the classifier) according to the ratio of false positives (i.e., fraction of incorrect scripts that are detected as correct by the classifier). The curve is moderately above the bisector, the area under this curve is equal to 0.585. Our classifier is therefore slightly better than a random classifier, but its performance is not very high; we make assumptions to explain these results in Section 7.

6.3 Study of the correlation with human notation

One of the objectives of our indicator is to reflect the learner's progress towards the solution to the problem, which means that its value is assumed to decrease as a script moves towards the solution. We therefore study the correlation between the value of our indicator under the parameter $\bar{\theta}$ for a given script and the matching correction, and the score assigned to this script by a human tutor.

This study is performed using a second dataset obtained after the final exam of the course described in Section 5. Of the 166 students enrolled in this course, 163 participated in the final exam where learners had to produce a script addressing a given problem. A teacher then assigned a score out of 10 to each of the 163 scripts. After eliminating grammatically incorrect scripts, this second dataset comprises 105 scripts.

Figure 4 shows, for each script, the value of the indicator according to the score assigned by the teacher. This graph shows a negative correlation: the (non-linear) Kendall correlation has a value equal to -0.319 (p value less than

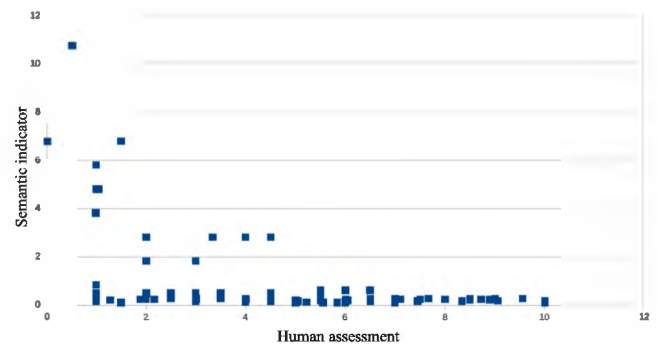


Figure 4: Value of our indicator according to score assigned by a human tutor.

$2.7 \cdot 10^{-6}$), the (linear) Pearson correlation having a value of -0.446 (p value equal to $1.9 \cdot 10^{-6}$). We therefore have a correlation between the score given by a human tutor and the value of our indicator. This correlation is not extremely high, but it may be sufficient to automatically estimate a student's progress towards solving a problem.

7. DISCUSSION

Our indicator is able to distinguish a semantically correct script from a semantically incorrect script, but its performance is slightly better than that of a random classifier only. A hypothesis to explain these results stems from the heterogeneous nature of the learning dataset, especially regarding the size of the scripts it contains. Indeed, the distance calculated by our indicator tends to increase with each token of a script. So the longer a script is, the more likely it is that the indicator will return a high value (even if the script approaches the solution), while a short and incorrect script is evaluated with a low value due to the low number of tokens.

The correlation study carried out on the second set of data gives indications about a certain capability of generalization of our approach. Indeed, unlike the test dataset extracted from the first dataset, the set of scripts obtained after the terminal exam corresponds to an exercise that is missing from the learning dataset. Therefore, our method seems to apply to exercises that were not used during the learning phase. However, this correlation is not extremely high, and scripts that are very poorly evaluated by human tutors are assigned a value of our indicator that is almost zero. This can also be explained by the hypothesis formulated above, since almost empty and therefore incorrect scripts are poorly rated by the human teacher, while our indicator returns a low value. Tests should be carried out to investigate how to adjust the algorithm to process the indicator, in particular according to the number of tokens comprised in the script representing the solution to the problem.

Developments have been initiated to return this indicator to learners during the programming activity. A first visualization reflects, as a graphical gauge, the value of the indicator each time a learner executes a script. A second visualization shows the learner's progress in solving the problem from the successive values of the indicator for the same script; it gives the variation of the indicator in the form of a gradient of colors ranging from green (for a short distance) to red

(for a long distance). The student is thus provided with a real time and easy to interpret view of the evolution of the accuracy of his script from a semantic point of view. These tools to support learners will be experimented in the next academic year with a new group of first-year students from HEIT. This experimentation will also make it possible to repeat the work presented in this study using a new dataset, and thus to improve the quality of our indicator.

8. CONCLUSION

In this paper, we conducted a study to design an indicator whose objective is to act as a foundation for intelligent guidance systems and awareness tools intended for both learners and teachers during a practical activity dedicated to learning programming. While a large variety of research is focused on the syntactic quality of the code produced by learners to support these learning activities, our originality lies in studying the semantic quality of the code, i.e., in evaluating the degree to which a program solves a given problem.

Thus, our main contribution relates on the design of an indicator that reflects a learner's ability to solve a problem. To answer the first research question asked in the introduction, we adapted the Levenshtein distance: from two strings of *tokens* representing the abstract syntax tree of the corresponding programs, the indicator returns an estimated value of the edit distance between the two scripts. To train and test this indicator, we used a dataset composed of scripts produced by students in authentic learning situations. The tests revealed that when it comes to differentiating between semantically correct and incorrect scripts, our indicator has slightly better performance than a random classifier. On the other hand, we observed an inverse correlation between the value of the indicator and the score assigned by a human tutor: the higher the human score of a program formulating a solution to a problem, the smaller the distance between that program and the solution of the problem.

These encouraging results suggest the opportunity to develop new models and tools dedicated to the semantic analysis of programming learning. However, many improvements need to be explored to improve the quality of our indicator. In addition to big amount of data required to refine the parameters of our indicator, the generalization of our approach to different programming languages must be checked, as well as the consideration of more complex programs.

9. REFERENCES

- [1] A. Ahadi, S. Lal, J. Leinonen, A. Hellas, and R. Lister. Performance and consistency in learning to program. In *Proceedings of The Nineteenth Australasian Computing Education Conference*, pages 11–16, Geelong, 2017. ACM.
- [2] R. Aiouni, A. Bey, and T. Bensebaa. An automated assessment tool of flowchart programs in introductory programming course using graph matching. *Journal of e-Learning and Knowledge Society*, 12(2):141–150, 2016.
- [3] A. Alammary, A. Carbone, and J. Sheard. Implementation of a smart lab for teachers of novice programmers. In *Proceedings of The 14th Australasian Computing Education Conference*, pages 121–130, Melbourne, 2012. ACS.
- [4] K. Beijering, C. Gooskens, and W. Heeringa. Predicting intelligibility and perceived linguistic distance by means of the levenshtein algorithm. *Linguistics in the Netherlands*, 25(1):13–24, 2008.
- [5] A. Bey, P. Jermann, and P. Dillenbourg. A comparison between two automatic assessment approaches for programming: An empirical study on moocs. *Journal of Educational Technology & Society*, 21(2):259–272, 2018.
- [6] E. Carter and G. D. Blank. A tutoring system for debugging: status report. *Journal of Computing Sciences in Colleges*, 28(3):46–52, 2013.
- [7] P. Denny, A. Luxton-Reilly, and D. Carpenter. Enhancing syntax error messages appears ineffectual. In *Proceedings of The 2014 Conference on Innovation & Technology in Computer Science Education*, pages 273–278, Uppsala, 2014. ACM.
- [8] P. Ihantola, A. Vihavainen, A. Ahadi, M. Butler, J. Börstler, S. H. Edwards, et al. Educational data mining and learning analytics in programming: Literature review and case studies. In *Proceedings of The 2015 ITiCSE on Working Group Reports*, pages 41–63, Vilnius, 2015. ACM.
- [9] M. Karam, M. Awa, A. Carbone, and J. Dargham. Assisting students with typical programming errors during a coding session. In *Proceedings of The Seventh International Conference on Information Technology*, pages 42–47, Las Vegas, 2010. IEEE.
- [10] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8):707–710, 1966.
- [11] M. Saeli, J. Perrenet, W. M. Jochems, and B. Zwaneveld. Teaching programming in secondary school: A pedagogical content knowledge perspective. *Informatics in Education*, 10(1):73–88, 2011.
- [12] K. Sharma, P. Jermann, and P. Dillenbourg. Identifying styles and paths toward success in moocs. In *Proceedings of The 8th International Conference on Educational Data Mining*, pages 408–411, Madrid, 2015. IEDMS.
- [13] K. Sharma, K. Mangaroska, H. Trætteberg, S. Lee-Cultura, and M. Giannakos. Evidence for programming strategies in university coding exercises. In *Proceedings of The 13th European Conference on Technology Enhanced Learning*, pages 326–339, Leeds, 2018. Springer.
- [14] J. Spacco, P. Denny, B. Richards, D. Babcock, D. Hovemeyer, J. Moscola, and R. Duvall. Analyzing student work patterns using programming exercise data. In *Proceedings of The 46th ACM Technical Symposium on Computer Science Education*, pages 18–23, Kansas City, 2015. ACM.
- [15] A. Taherkhani and L. Malmi. Beacon-and schema-based method for recognizing algorithms from students' source code. *Journal of Educational Data Mining*, 5(2):69–101, 2013.
- [16] C. Watson, F. W. Li, and J. L. Godwin. Predicting performance in an introductory programming course by logging and analyzing student programming behavior. In *Proceedings of The 13th International Conference on Advanced Learning Technologies*, pages 319–323, Beijing, 2013. IEEE.

Exploring the Link Between Motivations and Gaming

Steven Dang
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15206
stevenda@cs.cmu.edu

Ken Koedinger
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15206
koedinger@cmu.edu

ABSTRACT

A student's ability to regulate their thoughts, emotions and behaviors in the face of temptation is linked to their task specific motivational goals and dispositions. Behavioral tasks are designed to strain a targeted resource to differentiate individuals through measures of their performance. In this paper, we explore how student behavior on differentially gamed learning material relates to estimates of student motivational goals and dispositions. We leverage observations of students in two different courses using an intelligent tutoring system over an entire academic year. We use a previously validated heuristic model of gaming detection to label instances of gaming. Each student's tendency to game is estimated separately using highly gamed and non-highly gamed sections of the course. Each estimate of student gaming is compared to pre-course self-reported measures of student motivations. Results indicate that in naturalistic settings, gaming on more challenging materials is less influenced by student motivations and potentially a result of adaptive learning behaviors. Similarly, student gaming estimates using only non-highly gamed material are significantly related to all targeted motivation measures. Implications and future directions are discussed.

Keywords

Motivation, Self-regulation, Gaming, Measurement, Intelligent Tutoring System

1. INTRODUCTION

Students in the US currently spend on average between 20-25 hours per academic year on standardized testing [29]. The largest cost of formal standardized tests is the cost of lost learning opportunities for students. Additionally, these formalized, high-stakes assessments also lead to a range of other systemic effects, such as reductions in topical coverage and cultures of teaching to the test, that result in negative impacts to student learning [1]. While the need for measurement of student performance at all levels is necessary for the continued improvement of educational institutions, there is a need to identify solutions that balance the need for information on institutional performance with the learning needs of the student.

Steven Dang and Kenneth Koedinger "Exploring the Link Between Motivations and Gaming" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 276 - 281

The increased prevalence of digital learning resources in schools has created an opportunity to explore an alternative solution to standardized testing. [26] demonstrated the viability of leveraging longitudinal observations of student performance in an intelligent tutoring system to match assessments of student mathematics aptitude from standardized exams. Similarly, [31], demonstrated how the educational design process can be extended to designing of educational games to produce game-based activities that produce valid assessments of student skill.

Student cognitive skill is only half of the formula for student success; they must also have the motivation to apply those skills diligently over time to achieve [11]. Currently there are inadequate instruments available for high stakes measurement of student motivational constructs [13].

Self-regulation related learner behaviors are linked to student motivations. The characteristics of the context when students demonstrate failures to self-regulate their learning behaviors can be informative of their motivational goals [28], their perceived value of the activity [15], and their beliefs about their self-efficacy [32]. Drawing on design principles of psychometric behavioral tasks, we believe we can identify contexts that sufficiently load on student self-regulation to measure student motivations. In this paper, we seek to explore the feasibility of leveraging observations of students' self-regulation as measured by gaming the system behaviors to measure student motivational goals and dispositions.

2. MOTIVATION

We define motivation as the orienting and invigorating impact on both behavior and cognition of prospective reward [9]. For this study, we focus on a set of well-defined goals and dispositions that have been shown to influence student motivation and achievement.

A student's interest in a domain will influence the subjective value any task from that domain. This perceived expected value from completing a task influences students' self-regulation decisions [15]. Students vary in their beliefs about their ability to successfully complete a task, their self-efficacy, and this difference in appraisal affects motivation to apply effort to a task [4]. Effort regulation describes the ability of students to motivate themselves and persevere on a task in the face of difficulty or failure [24]. Growth mindset captures the beliefs students have about the nature of intelligence and whether or not it is malleable [14]. As with self-efficacy, mindsets impact motivation through task appraisals. Student goals in academic tasks can be described using a two dimensional representation of mastery vs performance and approach versus avoidance [33]. Students with mastery approach goals set goals to learn any assigned knowledge and skills. Students with performance approach goals are motivated by a desire to perform better than their peers, while performance avoidant students are motivated by goals to avoid performing worse than their peers.

3. RELATED WORK

A significant body of prior work has focused on assessing moment-by-moment motivation through detectors of affect [10] and engagement [12,18]. However, work analyzing the link between fine-grained behavioral measures and motivational goals and dispositions is much more limited.

[27] created a rational model of student affect that leveraged a range of individual attributes including Big 5 personality measures and achievement goals. This work established the value of students' achievement goals on predicting moment by moment motivations as inferred by affect.

Several researchers attempted to identify task specific behaviors that rationally should be linked to achievement goals. [20] attempted to relate help-seeking behaviors while using an ITS to achievement goals. Researchers expected mastery-oriented students to be more likely to use a glossary/index resource, while performance-oriented students might tend to ask for hints from the tutor instead. No significant relationship between self-reported achievement goals and help-seeking behaviors was found. However, task achievement goals as predicted by choice of help resources did relate to learning outcomes as would be predicted by achievement goal theory.

[25] expanded on this work and attempted to relate task choice, where descriptions of each task were closely linked to corresponding achievement goals, to self-reported achievement goals and learning outcomes. In this work, task achievement goals as inferred by task choice predicted learning outcomes for the lesson but did not align with self-reported achievement goals. However, self-reported achievement goals were more predictive of course outcomes. Researchers speculated that self-reported goals might reflect an average tendency to be motivated by particular goals over a range of tasks within the domain and thus explaining alignment with more aggregated measures such as course outcomes.

Gaming the system, a pattern of behavior where students abuse the design of the learning environment to answer a particular question, is a well-documented behavior that has been linked to poor learning outcomes [2]. In [3], the authors test the relationship between a range of student motivations and gaming the system behaviors across two different ITS's. The study results supported a link between gaming behaviors and some motivational measures but not others. One of the strongest results indicated that student's attitudes and interest towards the domain was related to observed gaming frequency. There was also strong support for a link between experiences of frustration and gaming as well as a lack of drive to motivate themselves on tasks in general as well as in the face of challenge. The results demonstrated mixed or weak support for a relationship with growth mindset and perceptions of the helpfulness of the ITS help resources. Interestingly, the researchers failed to identify a relationship between observed gaming and performance goals, though the performance goal measures were not drawn from validated achievement goal instruments. Furthermore, this study used strictly observed gaming frequencies. Subsequent work has identified the joint role of contextual and student factors in explaining gaming behaviors [19,21].

In this paper, we seek to answer two main research questions. Research Question #1: How does the relationship between gaming and measures of motivation differ when gaming estimates are derived from either raw observations of gaming or using random effects models that account for both student-level and contextual variation. Research Question #2: How does student performance

on educational content with varying degrees of gaming frequency relate to their different motivational goals and dispositions?

4. THE DATASET

For this study, we used a dataset drawn from [16] that was collected as part of a year-long study [6] in a suburban middle school in a mid-atlantic state.

The students used the Carnegie Learning Cognitive Tutor software (CogTutor). The CogTutor software provides adaptive instruction based on a fine-grained skill representation of the domain. The application divides problems into steps that must be answered individually and each map to independent skills in the domain model. Student practice problems are selected according to whether they have demonstrated mastery of necessary skills. The instruction is also scaffolded, allowing students to request multiple levels of hints at every step of the problem, providing on-demand problem scaffolding that provides increasingly informative support to the students. The data logs generated by the software are transformed into the standard learning data format specified by [16] before being utilized in this analysis. This format specifies how long students spend on every interaction, whether the action was correct, incorrect, or a hint, and what skill is associated with a specific problem step. Each interaction is represented as a single student transaction in the dataset, which includes over 2M such transactions across all students analyzed.

The dataset includes 189 students across 7 pre-algebra classes and 5 geometry classes. The population is predominantly white/Caucasian with only 2% of the sample being non-white. 56% of the students are female and 22% received either free or reduced lunch. The classes used the tutor for an entire academic year with an average of close to two class periods per week on the tutor. While the original dataset included 240 students across both of these courses, students with incomplete grade and survey responses were eliminated.

Additionally, some students and curricular sections were excluded due to having low observations in the data. The median student was observed during at least 40 sessions. However, 18 students were eliminated because EDA indicated these students as different from most others. The excluded students were observed in less than 20 class sessions and completed on average 780 total interactions with the system over the course of the year. By contrast, the median student completed about 15k transactions over the course of 40 sessions on average. These students were excluded from analysis because they appear unengaged and/or unmotivated, but there are so few observations of their behavior that drawing any conclusions from limited data is more prone to unobserved confounds.

Similarly, transactions from 31 sections are excluded from the dataset because they were observed with less than 6 students completing any work in the section. These sections are excluded because such sections might be measurements of only the fastest working or highest achieving students, thus introducing a bias to observations of gaming within those sections.

4.1 Motivational Measures

In addition to fine-grained student log data, several survey measures were collected at the beginning of the course to measure students' pre-course motivational goals and dispositions. Each scale utilized was drawn from well-validated instruments. Survey measures include scales for interest in math [17], self-efficacy [5], effort regulation [24], growth mindset [8] and achievement goals [33]. Questions from each scale are included in Figure 1 below. Each question was answered using a 5-point Likert rating, and responses

for each scale were summed to represent students' motivation along each dimension.

Interest in Math	
1.	Math is practical for me to know
2.	Math helps me in my daily life outside of school
3.	It is important to me to be a person who thinks mathematically
4.	Thinking mathematically is an important part of who I am
5.	I enjoy the subject of math
6.	I like math
7.	I enjoy doing math
8.	Math is exciting for me
Self-Efficacy	
1.	I am confident that I will do well in math class
2.	I expect to do well in math
3.	I am confident that I can learn future math concepts
4.	Considering the difficulty of this course, I think I will do well in mathematics in the future
5.	I am confident that I will do an excellent job on future math problems.
Effort Regulation	
1.	I often feel so lazy or bored when I do homework for math class that I quit before I finish what I planned to do.
2.	I work hard to do well in math class even if I don't like what we are doing.
3.	When class work is difficult, I give up or only study the easy parts.
4.	Even when math class assignments are dull and uninteresting, I manage to keep working until I finish.
Growth Mindset	
1.	You have a certain amount of intelligence and you really can't do too much to change it.
2.	Your intelligence is something about you that you can't change very much.
3.	You can learn new things, but you can't really change your intelligence.
4.	No matter who you are, you can change your intelligence a lot.
5.	You can always greatly change how intelligent you are.
6.	No matter how much intelligence you have, you can always change it quite a bit.
Achievement Goals	
1.	My aim is to completely master the material presented in this unit.
2.	In this unit, I am striving to do well compared to other students.
3.	In this unit, my goal is to learn as much as possible.
4.	In this unit, my aim is to perform well relative to other students.
5.	In this unit, my goal is to avoid performing poorly compared to others.
6.	I am striving to understand the content of this unit as thoroughly as possible
7.	My goal is to perform better than the other students in this unit
8.	In this unit, I am striving to avoid performing worse than others.
9.	In this unit, my aim is to avoid doing worse than other students.

Figure 1. Motivational survey inventory

4.2 Gaming the System Behaviors

We used the heuristic model of gaming behaviors introduced by [22] as this model appeared to produce better kappa on unseen data from across multiple systems including the CogTutor. Using this model, individual transactions were labeled according to a taxonomy that captures a range of relevant behaviors such as thinking before a hint request, spending time reading hint requests, and variations of guessing behaviors. Transactions are labeled as gaming if they are a member of a set of subsequent transactions that matches one of the thirteen identified patterns by [22] and shown in Figure 2. The patterns encode two primary types of gaming: guessing and hint abuse. Guessing patterns include placing the same answer incorrectly into multiple available answer slots and answering the same question rapidly with very small

changes in the answer across attempts. Hint abuse patterns include not stopping to think about multiple subsequent errors before requesting help and rapidly requesting hints to seek a bottom-out hint, which in the CogTutor environment is simply the answer to the problem step given as the second or third hint.

Transactions are rolled-up into student steps, where each student step encapsulates metadata about all the transactions associated with a problem step until a correct answer is reached. Each student step is labeled as gamed if any transaction associated with the step was also labelled gamed. The resulting student step data was utilized to calculate student and content gaming frequencies.

The overall dataset included 3.5% gamed student steps. These numbers align reasonably well with gaming frequencies observed in prior work on CogTutor data. [2] found students gaming the system about 3% of the time based on in-classroom human observations. [23] found a slightly higher overall gaming frequency of 6.8% in their dataset utilizing the same detection model as used here. However, this deviation isn't so different that it is due to significant unobserved differences in the populations.

incorrect → [guess] & [same answer/diff. context] & incorrect
incorrect → [similar answer] [same context] & incorrect → [similar answer] & [same context] & attempt
incorrect → [similar answer] & incorrect → [same answer/diff. context] & attempt
[guess] & incorrect → [guess] & [diff. answer AND/OR diff. context] & incorrect → [guess] & [diff. answer AND/OR diff. context & attempt
incorrect → [similar answer] & incorrect → [guess] & attempt
help & [searching for bottom-out hint] → incorrect → [similar answer] & incorrect
incorrect → [same answer/diff. context] & incorrect → [switched context before correct] & attempt/help
bug → [same answer/diff. context] & correct → bug
incorrect → [similar answer] & incorrect → [switched context before correct] & incorrect
incorrect → [switched context before correct] & incorrect → [similar answer] & incorrect
incorrect → [similar answer] & incorrect → [did not think before help] & help → incorrect (with first or second answer similar to the last one)
help → incorrect → incorrect → incorrect (with at least one similar answer between steps)
incorrect → incorrect → incorrect → [did not think before help request] & help (at least one similar answer between steps)

Figure 2. Patterns of Gaming

The CogTutor content is organized hierarchically into multiple units. Each unit consists of several sections that themselves have multiple skills to be learned. Each section has problems that are divided into highly granular steps which each are associated with at least one skill. We chose to group observations at the section level to capture differences across the curriculum with sufficient resolution while having sufficient observations across students to make reasonable estimates of gaming frequency. The data included 206 sections with a mean gaming frequency of 1.95% and a standard deviation of 1.7%. A number of sections were found to have 0 observed gaming, while there was one extreme outlier section with a frequency of 12.12%.

Unlike in prior work, [2], no students were found to have never gamed throughout the year. The average student was observed gaming 3.66% of the time with a standard deviation of 1.16%. The minimum observed gaming frequency for students was 1.98% while the maximum observed was 11.95%.

4.3 Comparing measures of gaming

In this study, we generate four estimates of student gaming and compare these estimates of student gaming frequency to each motivational measure using partial correlations. In the partial

correlations, we control for gender, ethnicity, and free/reduced lunch status.

- (1) $\theta_{ObservedGaming} = \frac{xNumGamed}{N_{TotalSteps}}$
- (2) $P(Gamed) \sim (1 | Student) + (1 | Section)$
- (3) $\theta_{Gaming} = e^{\theta_{student}}$

To investigate RQ1, we calculate student gaming using frequencies calculated using only raw observations for each student as shown in Eq 1. We also predict gaming on each step using a random effects model with a random effect for student and section as shown in Eq 2. The model is fit over all observed student steps and the student gaming is found by calculating the exponential of the fitted random intercept, $\theta_{student}$, for each student as shown in Eq 3. To investigate RQ2, we divide the data into two subsets, hard sections and non-hard sections. We set an 80% quantile cutoff of 3.06% gaming frequency for each section to identify non-highly gamed sections. There were 164 non-highly gamed sections and 41 highly gamed sections. Again, the random effects model in Eq 2 was used for each data subset to estimate student gaming.

For RQ1, we expect student gaming estimates from the random effects model to better correlate with motivation because the model takes into account variance in gaming due to sections, which may not be observed for all students, as well as accounting for statistical noise due to sampling of a rare event.

For RQ2, we investigate the hypothesis informed by design principles of psychometric behavioral tasks. Measuring a targeted construct requires straining the resource and identifying a metric upon which to differentiate subject performance. Therefore, we expect estimates of student gaming using only highly gamed sections will have a more significant relationship with motivational variables compared to data without highly gamed sections.

5. RESULTS

The results of the partial correlation analysis are shown in Tables 1.1 and 1.2. The first row of both tables present evidence contrary to the results from [3]. Prior research found correlations with math interest, effort regulation, and growth mindset using only averages of observed gaming. However, in this dataset, only interest in the subject is related to gaming behaviors, and no other motivational measure has a significant correlation with student's gaming frequency.

On the other hand, the second row reflects correlations with student gaming estimated using a random effects model fitted with all of the data. In general, more motivational measures are correlated with these gaming estimates than those derived from the raw observations, which supports the hypothesis for RQ1. Comparing these results to [3], there are no direct measures of frustration, however it is possible that self-efficacy mediates whether student's experience of frustration explaining the correlation. Growth mindset is found to be marginally significant, which further bolsters the previous mixed evidence for a link between mindsets and average student gaming.

There are two cells where these correlations do not seem to agree with prior research. Effort regulation is expected to be correlated both as a matter of face validity as well as because prior research found a relationship between gaming and students' drive to persevere on academic work.

The link between achievement goals and gaming are mixed. In [3], the authors assessed performance goals using questions such as, "If

you had your choice, what kind of extra-credit projects would you most likely do". It is unclear how this question maps to achievement goals, however, performance approach goals are not significant as might be extrapolated from prior work. On the contrary, mastery approach and performance avoidance goals are correlated with gaming. This relationship is rationally derived from the theory on self-regulation and motivation, but not predicted by specific prior work. Overall, the random effects model yielded a significant relationship to more motivational constructs than gaming estimates from raw observations.

Table 1.1 Correlations with Motivation Measures

Data Subset	Math Interest	Self Efficacy	Effort Reg.	Growth Mindset
Observed	-0.22**	-0.10	-0.11	0.00
All	-0.17*	-0.16*	-0.11	-0.14(.)
High Gaming	-0.19*	-0.14(.)	-0.10	-0.11
Low Gaming	-0.16*	-0.19*	-0.16*	-0.14*

(.) -0.10 > p ≥ 0.05, * - p < 0.05, ** - p < 0.01, *** - p < 0.001

Table 1.2 Correlations with Achievement Goals

Data Subset	Mastery Approach	Performance Approach	Performance Avoidance
Observed	-0.03	-0.01	-0.05
All	-0.20**	-0.10	-0.15*
High Gaming	-0.14(.)	-0.08	-0.11
Low Gaming	-0.25***	-0.14(.)	-0.21**

(.) -0.10 > p ≥ 0.05, * - p < 0.05, ** - p < 0.01, *** - p < 0.001

The results from estimating gaming using only highly gamed sections, row #3, are contrary to what is expected. Many of the correlations that appear when using all of the data, are weakened or not significant when using only the hardest questions. While the loss of significance with some constructs could be an artifact of random sampling from the full dataset, this does not explain the results seen in the bottom row. When estimating gaming using only non-highly gamed sections, correlations arise with every available motivational construct as seen in the fourth row. This is an unlikely consequence of sampling from the population and supports the idea that student gaming performance on highly gamed questions is introducing additional noise to the available signal in the rest of the data. Thus, the evidence points towards student gaming behaviors in the non-highly gamed sections as being more informative of student motivations than behaviors in the highly-gamed sections where self-regulation is under greater strain.

5.1 Exploring High-gamed sections

One possible explanation for this counterintuitive result is that gaming in highly gamed sections could be due to poorly designed content instead of cognitively challenge questions. To quantify average section difficulty, we calculate the ratio of the number of steps where student's first transaction is either a hint request or an error to the total number of student steps observed for each section (the assistance score). The highly gamed sections do in fact appear to be more difficult sections. The average highly-gamed section has 18.3% assistance steps with a standard deviation of 5.2%. The easiest section in this subset was observed with 9.9% assistance. By contrast, the average non-highly-gamed section consisted of 9.6% assistance steps with a standard deviation of 4.1%. Therefore,

students required less overall assistance on the majority of non-highly-gamed sections than the easiest of the highly-gamed sections.

If the content, was in fact challenging to students, perhaps the average questions were beyond the abilities of students to answer. If students are unmotivated to try on difficult problems, then students should be most likely to game on the first encounter with a particular skill. As experience with a skill goes up, perceived difficulty should go down, thus monotonically reducing the probability of gaming with each practice opportunity. However, if a skill is beyond a student's ability and the student is sufficiently motivated to attempt to learn, then the student should be less likely to game on the first encounter of a skill than on subsequent encounters. In this case, after first encountering a skill, students are able to assess that the skill is too far beyond their abilities. On subsequent encounters with the skill, it is an adaptive behavior to abuse help resources or apply other gaming strategies to acquire the answer. Prior work has similarly found that not all gaming is harmful to learning [2], for instance students may be using hints as a form of worked example [30].

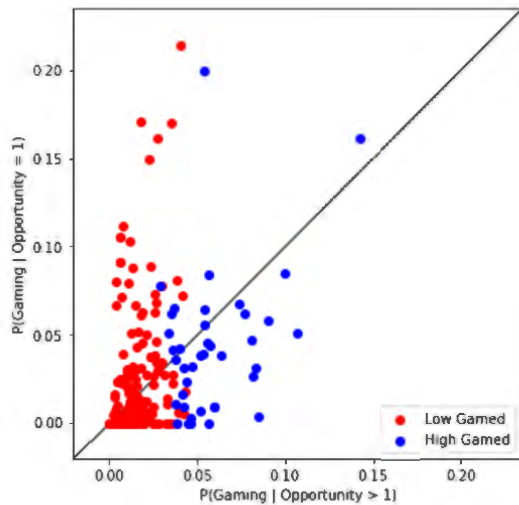


Figure 3. Gaming on 1st opportunity vs subsequent

In Figure 3, the observed gaming on 1st opportunities and all subsequent opportunities are compared for each section. Students in most highly gamed sections, in blue, are more likely to game of subsequent opportunities than the first. 70.7% of highly gamed sections share this characteristic as compared to 46.9% of non-highly gamed sections. This evidence, in addition to the lack of correlation between gaming on high-gamed sections and student's effort regulation supports the rational that gaming is sometime a somewhat desirable adaptive behavior for students and observations of gaming and in these sections should be treated differently than in other sections. In fact, the ratio of gaming on first opportunity to gaming on subsequent opportunities might be a valuable measure to incorporate into future motivational measurement models.

6. DISCUSSION

In this study, we demonstrate that leveraging random effects models to cope with statistical noise in observations of student's tendency to game on any given section better estimates student's gaming as related to their motivational goals and dispositions. Additionally, we provide initial evidence towards a measurement model of student's motivational goals and dispositions by leveraging observations of gaming. Results indicate a relationship

between gaming and motivation that involves an interaction with the difficulty level and prior experience with a problem.

Several correlations with gaming estimates appear contrary to prior research and merit further analysis. The significant correlation with both mastery approach and performance avoidance disagrees with the results found by [3]. This disagreement could be due to the independence of achievement goals from each other, where gaming may be driven by an aggregate motivation of all achievement goals. More analysis is necessary to bridge this seeming contradiction and understand how patterns of gaming across problems of varying difficulty and prior experience might support an interpretation of gaming as indicative of different achievement goal profiles.

Gaming frequency was leveraged as a proxy measure for a range of unencoded difficulty factors. While this includes factors such as poor classroom instruction or a poorly designed cognitive model, it also encapsulates difficulty of individual problem-steps. A natural next step would be to investigate how more detailed student skill models might improve estimates of perceived difficulty and corresponding enrich model understanding of the nuances of why students are gaming and how this relates to different motivational goals.

Prior work has also shown that on longer time-scales, motivational goals are not necessarily stable [7]. In this study, we looked for relationships between pre-course motivations and in-course gaming behaviors. For students with fluctuations in achievement goals or self-efficacy, the contexts in which such students tend to game or disengage from the lesson in other manners might similarly change. Further analysis is necessary to investigate whether variations in gaming over time are similarly reflective of variations in motivational goals and dispositions over time.

Furthermore, the study included a fairly large body of students, but the observations were still limited to a single school in a particular region of the country with limited ethnic and socio-economic status diversity represented in the sample. Such factors are known to be correlated with variations in the types and frequencies of gaming behaviors observed in the population [21]. As such, we exercise caution in extrapolating these relationships beyond this demographic group without further validation.

Nonetheless, the results presented in this work lay the groundwork for further investigation into measurement models of motivational goals and dispositions that leverage an understanding of the contexts that strain students' self-regulation. Such unobtrusive measurement models hold the keys to a future where schools can better utilize instructional time that is currently occupied by standardized test and test-specific preparation while still receiving the student, and class-level performance measures necessary to support continuous improvement.

7. ACKNOWLEDGMENTS

The research reported here was supported, in whole or in part, by the Institute of Education Sciences, U.S. Department of Education, through grant R305B150008 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent the views of the Institute or the U.S. Department of Education.

8. REFERENCES

- [1] Wayne Au. 2016. High-Stakes Testing and Curricular Control: A Qualitative Metasynthesis. 36, 5: 258–267.
- [2] Ryan Shaun Baker, Albert T Corbett, Kenneth R Koedinger, and Angela Z Wagner. 2004. Off-task behavior in the cognitive tutor classroom. 383–390.

- [3] Ryan Baker. 2008. Why Students Engage in “Gaming the System” Behavior in Interactive Learning Environments. *Why Students Engage in “Gaming the System” Behavior in Interactive Learning Environments* 19, 2: 185–224.
- [4] Albert Bandura. 1997. Self-Efficacy: The Exercise of Control. Macmillan.
- [5] Albert Bandura. 2006. Guide for Constructing Self-Efficacy Scales. In *Self-Efficacy Beliefs of Adolescents*. 307–337.
- [6] Matthew L Bernacki and Steven Ritter. 2013. Hopewell 2011-2012. Dataset 613 in DataShop. Retrieved from <https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=613>.
- [7] Matthew L Bernacki, Vincent Aleven, and Timothy J Nokes-Malach. 2014. Stability and change in adolescents’ task-specific achievement goals and implications for learning mathematics with intelligent tutors. *Computers in Human Behavior* 37: 73–80.
- [8] Lisa S Blackwell, Kali H Trzesniewski, and Carol Sorich Dweck. 2007. Implicit Theories of Intelligence Predict Achievement Across an Adolescent Transition: A Longitudinal Study and an Intervention. *Child development* 78, 1: 246–263.
- [9] Matthew Botvinick and Todd Braver. 2015. Motivation and Cognitive Control: From Behavior to Neural Mechanism. *Annual Review of Psychology* 66, 1: 83–113.
- [10] Rafael A Calvo and Sidney D’Mello. 2010. Affect Detection: An Interdisciplinary Review of Models, Methods, and Their Applications. *IEEE Transactions on Affective Computing* 1, 1: 18–37.
- [11] Martin V Covington. 1992. *Making the Grade*. Cambridge University Press.
- [12] Sidney D’Mello, Ed Dieterle, and Angela Duckworth. 2017. Advanced, Analytic, Automated (AAA) Measurement of Engagement During Learning. *Educational Psychologist*: 1–20.
- [13] Angela L Duckworth and David Scott Yeager. 2015. Measurement matters assessing personal qualities other than cognitive ability for educational purposes. 44, 4: 237–251.
- [14] Carol Dweck. 2007. Mindset: The New Psychology of Success.
- [15] Jacquelynne S Eccles. 2005. Subjective task value and the Eccles et al. model of achievement-related choices. *Handbook of competence and motivation*: 105–121.
- [16] Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J. 2010. A Data Repository for the EDM community: The PSLC DataShop. In Romero, C., Ventura, S., Pechenizkiy, M., Baker, R.S.J.d. (Eds.) *Handbook of Educational Data Mining*. Boca Raton, FL: CRC Press.
- [17] Lisa Linnenbrink-Garcia, Amanda M Durik, AnneMarie M Conley, Kenneth E Barron, John M Tauer, Stuart A Karabenick, and Judith M Harackiewicz. 2010. Measuring Situational Interest in Academic Domains. *Educational and Psychological Measurement* 70, 4: 647–671.
- [18] Caitlin Mills, Sidney D’Mello, Nigel Bosch, and Andrew M Olney. 2015. Mind Wandering During Learning with an Intelligent Tutoring System. In *Artificial Intelligence in Education*. Springer, Cham, Cham, 267–276.
- [19] Kasia Muldner, Winslow Burleson, Brett Van de Sande, and Kurt VanLehn. 2011. An analysis of students’ gaming behaviors in an intelligent tutoring system: predictors and impacts. *User Modeling and User-Adapted Interaction* 21, 1-2: 99–135.
- [20] Christine Otieno, Rolf Schwonke, Ron Salden, and Alexander Renkl. 2013. Can Help Seeking Behavior in Intelligent Tutoring Systems Be Used as Online Measure for Goal Orientation? In *Proceedings of the Annual Meeting of the Cognitive Science Society* 35, 35.
- [21] Luc Paquette and Ryan S Baker. 2017. Variations of Gaming Behaviors Across Populations of Students and Across Learning Environments. In *International Conference on Artificial Intelligence in Education* 374-286.
- [22] Luc Paquette, Adriana de Carvahlo, Ryan Baker, and Jaclyn Ocuppaugh. 2014. Reengineering the Feature Distillation Process: A case study in detection of Gaming the System. In *Proceedings of the Secenth International Conference for Educational Data Mining*. London, UK.
- [23] Luc Paquette, Adriana M J A de Carvalho, and Ryan S Baker. 2014. Towards Understanding Expert Coding of Student Disengagement in Online Learning. In *Proceedings of the 36th Annual Cognitive Science Conference*, 1126-1131.
- [24] Paul R Pintrich. 1991. A Manual for the Use of the Motivated Strategies for Learning Questionnaire (MSLQ).
- [25] J E Richey, Matthew L Bernacki, and D M Belenky. Relating a Task-Based, Behavioral Measure of Achievement Goals to Self-Reported Goals and Performance in the Classroom. In *Proceedings of the Annual Meeting of the Cognitive Science Society* 36,36.
- [26] Steve Ritter, Ambarish Joshi, Stephen E Fancsali, and Tristan Nixon. 2013. Predicting Standardized Test Scores from Cognitive Tutor Interactions. In *Proceedings of the Sixth International Conference on Educational Data Mining*.
- [27] Jennifer Sabourin, Bradford Mott, and James C Lester. 2011. Generalizing Models of Student Affect in Game-Based Learning Environments. In *Affective Computing and Intelligent Interaction* (2nd ed.). Springer, Berlin, Heidelberg, Berlin, Heidelberg, 588–597.
- [28] Steven J Scher and Nicole M Osterman. 2002. Procrastination, conscientiousness, anxiety, and goals. *Psychology in the Schools* 39, 4: 385–398.
- [29] Council of the Great City Schools. 2015. Student Testing in America’s Great City Schools: An Inventory and Preliminary Analysis. 1–164.
- [30] Benjamin Shih, Kenneth R Koedinger, and Richard Sheines. 2008. A Response Time Model for Bottom-Out Hints as Worked Examples. 117–126.
- [31] Valerie J Shute. 2011. Stealth assessment in computer-based games to support learning. *Computer games and instruction* 55: 22, 503-524.
- [32] Christopher A Wolters. 2003. Understanding procrastination from a self-regulated learning perspective. *Journal of Educational Psychology*, 95: 179.
- [33] 2000. An Achievement Goal Theory Perspective on Issues in Motivation Terminology, Theory, and Research. *Contemporary Educational Psychology* 25, 1: 92–104.

Why Deep Knowledge Tracing has less Depth than Anticipated

Xinyi Ding
Lyle School of Engineering
Southern Methodist University
xding@smu.edu

Eric C. Larson
Lyle School of Engineering
Southern Methodist University
eclarson@lyle.smu.edu

ABSTRACT

Knowledge tracing allows Intelligent Tutoring Systems to infer which topics or skills a student has mastered, thus adjusting curriculum accordingly. Deep Knowledge Tracing (DKT) uses recurrent neural networks (RNNs) for knowledge tracing and has achieved significant improvements compared with models like Bayesian Knowledge Tracing (BKT) and Performance Factor Analysis (PFA). However, DKT is not as interpretable as other models because the decision-making process learned by recurrent neural networks is not wholly understood by the research community. In this paper, we critically examine the DKT model, visualizing and analyzing the behaviors of DKT in high dimensional space. We modify and explore the DKT model and discover that Deep Knowledge Tracing has some critical pitfalls: 1). instead of tracking each skill through time, DKT is more likely to learn an ‘ability’ model; 2) the recurrent nature of DKT reinforces irrelevant information that it uses during the tracking task; 3) an untrained recurrent network can achieve similar results to a trained DKT model, supporting a conclusion that recurrence relations are not properly learned and, instead, improvements are simply a benefit of projection into a high dimensional, sparse vector space. Based on these observations, we propose improvements and future directions for conducting knowledge tracing research using deep models.

Keywords

knowledge tracing, recurrent neural network, visualization

1. INTRODUCTION

Knowledge tracing has been investigated for decades. It allows Intelligent Tutoring Systems to infer which topics or skills a student has mastered, thus adjusting curriculum accordingly. Two widely used models are Bayesian Knowledge Tracing (BKT) [2] and Performance Factor Analysis (PFA)[11]. These models are designed in a way that each parameter has a semantic meaning. For example, the guess

and slip parameter in the BKT model reflect the probability that a student could guess the correct answer and make a mistake despite mastery of a skill, respectively. BKT attempts to explicitly model these parameters and use them to infer a binary set of skills as mastered or not mastered. In parallel with research in knowledge tracing models, deep neural networks have gained popularity in fields like Natural Language Processing and Computer Vision [3, 9]. Piech *et. al* proposed Deep Knowledge Tracing (DKT) [12], using recurrent neural networks for knowledge tracing. The DKT model achieves significantly improved results compared to BKT and PFA. However, its mechanisms are not well understood by the research community. That is, none of the parameters are mapped to a semantically meaningful measure which diminishes our ability to understand how DKT performs predictions. There have been some attempts to explain why DKT works well [8, 15], but these studies treat DKT model more like a black box, without studying the state space that underpins the recurrent neural network. In this work, we analyze and visualize the learned state space of the DKT model to better understand its mechanisms.

Recurrent neural networks can learn long range dependencies across many time steps. Long short term memory (LSTM) networks, gated-recurrent unit (GRU) networks, and numerous other variants enhance the vanilla RNNs in one way or another have achieved empirical success [6, 1, 5]. However, there are incredibly few works explaining what is happening under the hood. Karpathy *et al.* [7] provide a detailed analysis of the behaviors of recurrent neural network in language processing and find that some neurons are responsible for long range dependencies like quotes and brackets. We take a similar approach for analyzing the DKT model.

We aim to provide a better understanding of the DKT model and a more solid footing for using deep models for knowledge tracing. In this paper, we “open the box” of the DKT recurrent architecture, visualizing and analyzing the behaviors of the DKT model in a high dimensional space. We track activation changes through time and analyze the impact of each skill in relation to other skills. We modify and explore the DKT model, finding that some irrelevant information is reinforced in the recurrent architecture. Finally, we find that an untrained DKT model (with gradient descent applied only to layers outside the recurrent architecture) can be trained to achieve similar performance as a fully trained DKT architecture. Based on our analyses, we propose improvements and future directions for conducting knowledge

Xinyi Ding and Eric Larson "Why Deep Knowledge Tracing has less Depth than Anticipated" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 282 - 287

tracing with deep recurrent neural network models.

2. RELATED WORK

Bayesian Knowledge Tracing (BKT) [2] was proposed by Corbett *et al.* In their original work, each skill has its own model and parameters are updated by observing the responses (correct or incorrect) of applying a skill. Performance Factor analysis (PFA) [11] is an alternative method to BKT and it is believed to perform better when each response requires multiple skills. Both BKT and PFA are designed in a way that each parameter has its own semantic meaning. For example, the slip parameter of BKT represents the possibility of getting a question wrong even though the student has mastered the skill. These models are easy to interpret, but suffer from scalability issues and often fail to capture the dependencies between each skill because many elements are treated as independent to facilitate optimization.

Piech *et al.* recently proposed the Deep Knowledge Tracing model (DKT) [12], which exploits recurrent neural networks for knowledge tracing and achieves significantly improved results. Piesch *et al.* transformed the problem of knowledge tracing by assuming each question can be associated with a “skill ID”, with a total of N skills in the question bank. The input to the recurrent neural network is a binary vector encoding of skill ID for a presented question and the correctness of the student’s response. The output of the recurrent network is a length N vector of probabilities for answering each skill-type question correctly. The DKT model could achieve >80% AUC on the ASSISTmentsData dataset [4], compared with the BKT model that achieves 67% AUC. This is an exciting result because it demonstrates the possibility of using neural networks for knowledge tracing.

Despite the effectiveness of DKT model, its mechanism is not well understood by the research community. Khajah *et al.* investigate this problem by extending BKT [8]. They extend BKT by adding forgetting, student ability, and skill discovery components, comparing these extended models with DKT. Some of these extended models could achieve close results compared with DKT. Xiong *et al.* discover that there are duplicates in the original ASSISTmentsData dataset [15]. They re-evaluate the performance of DKT on different subsets of the original dataset. Both Khajah and Xiong’s work are black box oriented—that is, it is unclear how predictions are performed within the DKT model. In our work, we try to bridge this gap and explain some behaviors of the DKT model.

Trying to understand how DKT works is difficult because the mechanisms of RNNs are not totally understood even in the machine learning community. Even though the recurrent architecture is well understood, it is difficult to understand how the model adapts weights for a given prediction task. One common method used is to visualize the neuron activations. Karpathy *et al.* [7] provide a detailed analysis of the behaviors of recurrent neural network using character level models and find some cells are responsible for long range dependencies like quotes and brackets. They break down the errors and partially explain the improvements of using LSTM. We use and extend their methods, providing a detail analysis of the behaviors of LSTM in the knowledge tracing setting.

3. EXPERIMENT

To investigate the DKT model, we perform a number of analyses based upon the activations within the recurrent neural network. We also explore different training protocols and clustering of the activations to help elucidate what is learned by the DKT model.

3.1 Experiment setup

In our analyses, we use the “ASSISTmentsData 2009-2010 (b) dataset” which is created by Xiong *et al.* after removing duplicates [15]. Like Xiong *et al.*, we also use LSTM units for analysis in this paper. Because we will be visualizing specific activations of the LSTM, it is useful to review the mathematical elements that comprise each unit. An LSTM unit consists of the following parts, where a sequence of inputs $\{x_1, x_2, \dots, x_T\} \in \mathcal{X}$ are ideally mapped to a labeled output sequence $\{y_1, y_2, \dots, y_T\} \in \mathcal{Y}$. The prediction goal is to learn weights and biases (W and b) such that the model output sequence $\{h_1, h_2, \dots, h_T\} \in \mathcal{H}$ is as close as possible to \mathcal{Y} :

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (3)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

Here, σ refers to a logistic (sigmoid) function, \cdot refers to dot products, $*$ refers to element-wise vector multiplication, and $[,]$ refers to vector concatenation. For visualization purposes, we log the above 6 intermediate outputs for each input during testing and concatenate these outputs into a single “activation” vector, $a_t = [f_t, i_t, \tilde{C}_t, C_t, o_t, h_t]$. In the DKT model, the output of RNN, h_t is connected to an output layer y_t , which is a vector with the same number of elements as skills. We can interpret each element in y_t as an

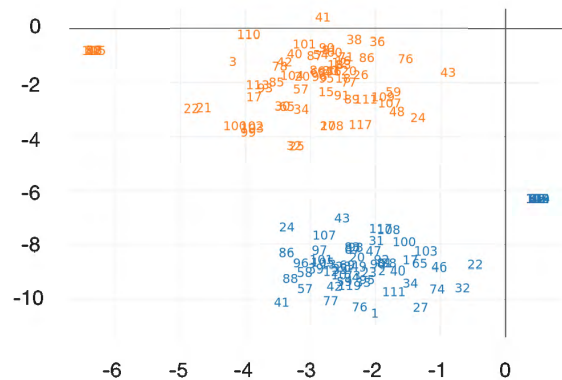


Figure 1: First two components of T-SNE of the activation vector for first time step inputs. Numbers are skill identifiers, blue for correct input, orange for incorrect input

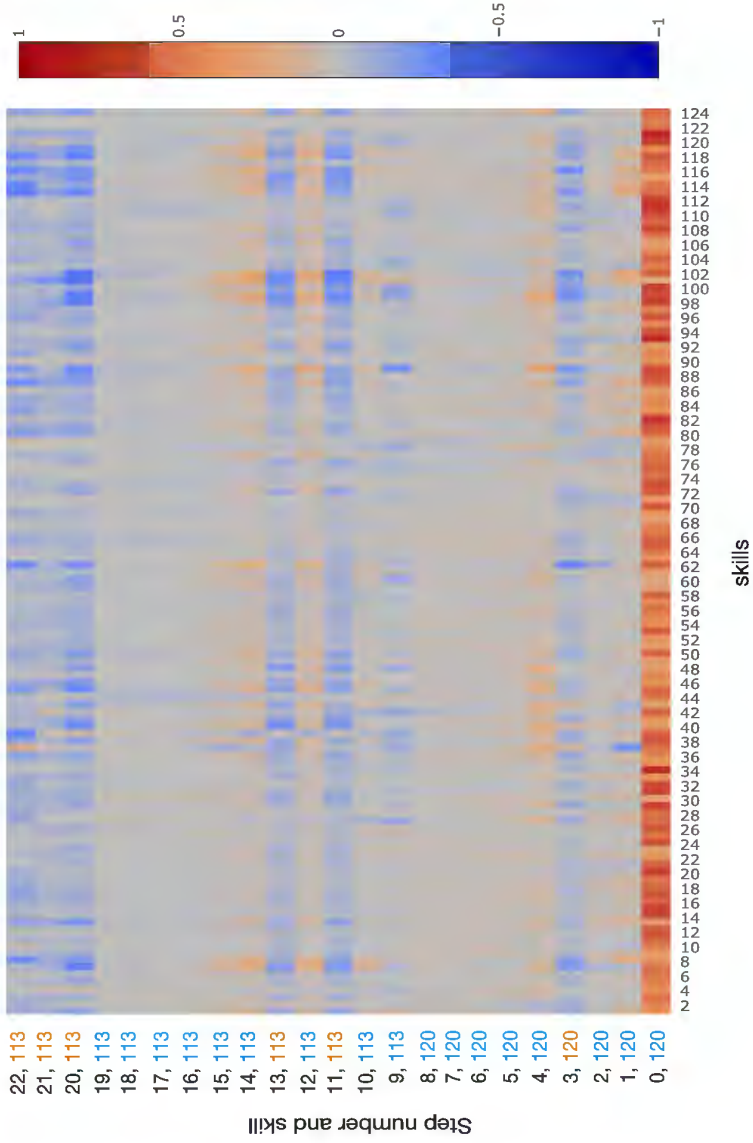


Figure 2: The prediction changes for one student, 23 steps, correct input is marked blue, incorrect input is marked orange

estimate that the student would answer a question from each skill correctly, with larger positive number denoting that the student is more likely to answer correctly and more negative numbers denoting that the student is unlikely to respond correctly. Thus, a student who had mastered all skills would ideally obtain an y_t of all ones. A student who had mastered none of the skills would ideally obtain an y_t of all negative ones.

Deep neural networks usually work in high dimensional space and are difficult to visualize. Even so, dimensionality reduction techniques can help to identify clusters. For example, Figure 1 plots the first two reduced components (using t-SNE [10]) of the activation vector, a_t , at the first time step ($t = 0$) for a number of different students in the ASSISTmentsData. The numbers in the plot are skill identifiers. We use color blue to denote a correct response and the color orange to denote an incorrect response. From reducing the dimensionality of the a_t vector for each student, we can see that the activations show a distinct clustering between whether the questions were answered correctly or incorrectly. We might expect to observe sub-clusters of the skill identifiers within each of the two clusters but we do not. This observation supports the hypothesis that correct and incorrect responses are more important for the DKT model than skill identifiers. However, perhaps this lack of sub-clusters is inevitable because we are only visualizing the activations after one time step—this motivates the analysis in the next section.

3.2 Skill relations

In this section, we try to understand how the prediction vector of one student changes as a student answers more questions from the question bank. Figure 2 plots the prediction difference (current prediction vector - previous prediction vector) for each question response from one particular student (steps are displayed vertically and can be read sequentially from bottom to top). The horizontal axis denotes the skill identifier and the color of the boxes in the heatmap denote the change in the output vector y_t . The initial row in the heatmap (bottom) is the starting values for y_t for the first input. As we can see, if the student answers correctly, most of the y_t values increase (warm color). When an incorrect response occurs, most of the predictions decreases (cold color). This makes intuitive sense. We expect a number of skills to be related so correct responses should add value and incorrect responses should subtract value. We can further observe that changes in the y_t vector diminish if the student correctly or incorrectly answers a question from the same skill several times repeatedly. For example, observe from step 14 to step 19, where the student correctly answers questions from skill #113—eventually the changes in y_t come to a steady state. However, occasionally, we can also notice, a correct response will result in decreases in the prediction vector (observe step 9). This behavior is difficult to justify from our experience, as correctly answering a question should not decrease the mastery level of other skills. Yeung *et al.* have similar findings when investigating single skills [16]. Observe also that step 9 coincides with a transition in skills being answered (from skill #120 to #113). Even so, it is curious that switching from one skill to another

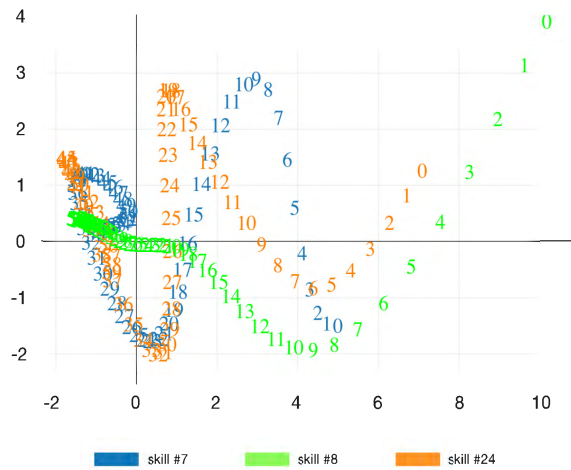


Figure 3: Activation vector changes for 100 continuous correctness of randomly picked 3 skills

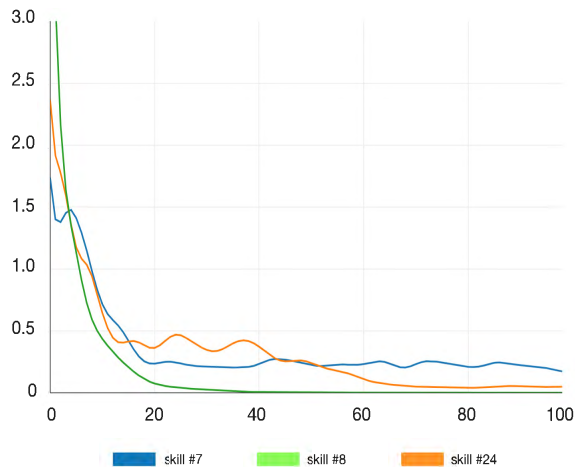


Figure 4: Activation vector difference of randomly picked 3 skills through time

other would decrease values in y_t even when the response is correct. From this observation, one potential way to improve the DKT model could be adding punishment for such unexpected behaviors (for example, in the loss function of the recurrent network).

3.3 Simulated data

From the above analysis, we see from step 14 to step 19, the student correctly answers question from skill #113 and the changes in y_t diminish—perhaps an indication that the vector is converging. Also, from Figure 2, we see that for each correct input, most of the elements of y_t increase by some margin, regardless of the input skill. To have a better understanding of this convergence behavior, we simulate how the DKT model would respond to an *Oracle Student*, which will always answer each skill correctly. We simulate how the model responds to the *Oracle Student* correctly answering 100 questions from one skill. We repeat this for three randomly selected skills.

We plot the convergence of each skill using the activation vector a_t reduced to a two-dimensional plot using t-SNE (Figure 3). The randomly chosen skills were #7, #8, and #24. As we can see, each of the three skills starts from a different location in the 2-D space. However, they each converges to near the same location in space. In other words, it seems DKT is learning one “oracle state” and this state can be reached by practicing any skill repeatedly, regardless of the skill chosen. We verified this observation with a number of other skills (not shown) and find this behavior is consistent. Therefore, we hypothesize that DKT is learning a ‘student ability’ model, rather than a ‘per skill’ model like BKT. To make this observation more concrete, in Figure 4 we plot the euclidean distance between the current time step activation vector, a_t , and the previous activations, a_{t-1} , we can see the difference becomes increasingly small after 20 steps. Moreover, the euclidean distance between each activation vector learned from each skill becomes extremely small, supporting the observation that not only is the y_t output vector converging, but all the activations inside the LSTM network are converging. We find this behavior curious because it means that the DKT model is not remembering what skill was used to converge the network to an ‘oracle state.’ Remembering the starting skill would be crucial for predicting future performance of the student, yet the DKT model would treat every skill identically. We also analyzed a process where a student always answers responses incorrectly and found there is a similar phenomenon with convergence in an anti-oracle state.

Figure 5 shows the skills prediction vector after answering correctly 20 times in a row. We can see the predictions of most skills are above 0.5, regardless of the specific practice skill used by the *Oracle Student*. Now, we can safely say that the DKT model is not really tracking the mastery level of *each skill*, it is more likely learning an ‘ability model’ from the responses. Once a student is in this oracle state, DKT will assume that he/she will answer most of the questions correctly from any skill. We hypothesize that this behavior could be mitigated by using an “attention” vector during the decoding of the LSTM network [13]. Self attention in recurrent networks decodes the state vectors by taking a weighted sum of the state vectors over a range in the sequence (weights are dynamic based on the state vectors). For DKT, this attention vector could also be dynamically allocated based upon the skills answered in the sequence, which might help facilitate remembering long-term skill dependencies.

3.4 Temporal impact

RNNs are typically well suited for tracking relations of inputs in a sequence, especially when the inputs occur near one another in the sequence. However, long range dependencies are more difficult for the network to track [13]. In other words, the predictions of RNN models will be more impacted by recent inputs. For knowledge tracing, this is not a desired characteristic. Consider two scenarios as shown below: For each scenario, the first line is the skill numbers and the second line are responses (1 for correctness and 0 for incorrectness). Both two scenarios have the same number of attempts for each skill (4 attempts for skill #9, 3 attempts for skill #6 and 2 attempts for skill #24). Also, the ordering of correctness within each skill is the same (e.g., 1, 0, 0, 0

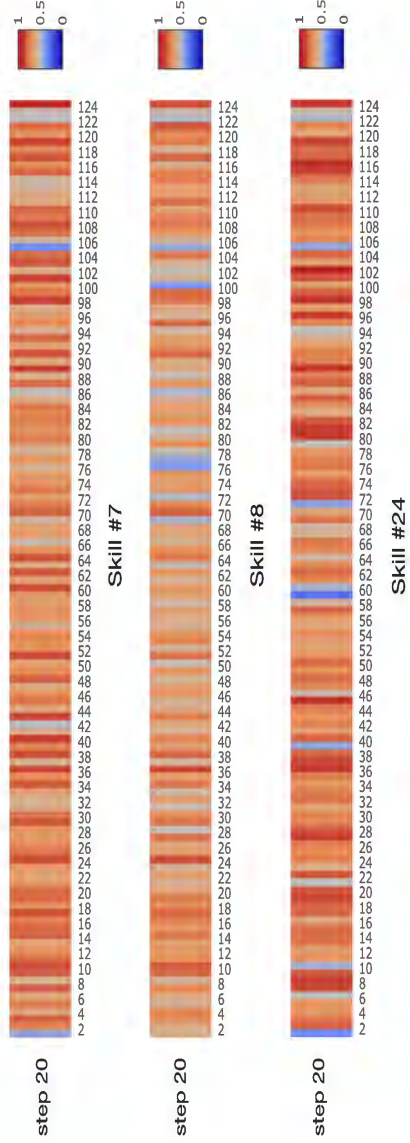


Figure 5: Prediction vector after 20 steps for skill #7, #8, #24

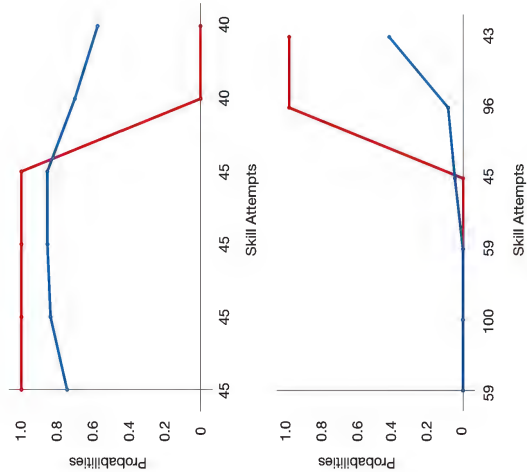


Figure 6: DKT predictions from two different students. The blue line is the prediction of correctness from DKT. The red line is the actual response correctness(1 or 0).

for skill #9).

Scenario #1									
Skill ID	6	6	9	9	9	9	24	24	6
Correct	1	1	1	0	0	0	0	0	1

Scenario #2									
Skill ID	9	9	9	9	6	6	6	24	24
Correct	1	0	0	0	1	1	1	0	0

For models like BKT, there is a separate model for each skill. Thus, the relative order of different skills presented has no influence, as long as the ordering within each skill remains the same. In other words, for each skill the ordering of correct and incorrect attempts remains the same, but different skills can be shuffled into the sequence. For BKT, it will learn the same model from these two scenarios, but it may not be the case for DKT. The DKT model is more likely to predict incorrect response after seeing three incorrect inputs

in a row because it is more sensitive to recent inputs in the sequence. This means, for the first scenario, first attempt of skill #24 (in bold) will be more likely predicted incorrect because it follows three incorrect responses. For the second scenario, first attempt of skill #24 (in bold) is more likely to be predicted correct. Thus the DKT model might perform differently on the given scenarios.

Figure 6 gives two typical excerpts from the real dataset for two students. In the top example, after several correct inputs, the DKT model has a high probability of predicting the next item correct, regardless of the skill (70%). Similarly, in the bottom example, after several incorrect inputs, the DKT model has a low probability of predicting the next item correct (8%), regardless of the skill. That means, if a student has mastered an easy skill previously but then fails three attempts of more difficult exercises, the DKT would predict that the student would also fail the already mastered skill. We are only giving two samples here due to limited space, but this kind of behavior is universal across students, which we will talk more next. Again, we hypothesize that this behavior could be mitigated by using an “attention” vector that allows the DKT to use the whole weighted history as additional inputs.

Table 1: Area under the ROC curve

	PFA	BKT	DKT	DKT (spread)	DKT (untrained)
09-10 (a)	0.70	0.60	0.81	0.72	0.79
09-10 (b)	0.73	0.63	0.82	0.72	0.79
09-10 (c)	0.73	0.63	0.75	0.71	0.73
14-15	0.69	0.64	0.70	0.67	0.68
KDD	0.71	0.62	0.79	0.76	0.76

Table 2: Square of linear correlation (r^2) results

	PFA	BKT	DKT	DKT (spread)	DKT (untrained)
09-10 (a)	0.11	0.04	0.29	0.15	0.25
09-10 (b)	0.14	0.07	0.31	0.14	0.26
09-10 (c)	0.14	0.07	0.18	0.14	0.15
14-15	0.09	0.06	0.10	0.08	0.09
KDD	0.10	0.05	0.21	0.17	0.17

Khajah *et al.* also alluded to this recency effect in [8]. In this paper, we examine this phenomenon in a more quantitative way. We shuffle the dataset in a way that keeps the ordering within each skill the same, but spreads out the responses in the sequence. This change should not change the prediction ability of models like BKT. The results are shown in Table 1 and Table 2 using standard evaluation criteria for this dataset. All results are based on a five-fold cross validation of the dataset. When comparing DKT on the original dataset to the “spread out” dataset ordering, we see that the relative ordering of skills has significant negative impact on the performance of the model. From these observations, we see the behaviors of DKT is more like PFA which counts prior frequencies of correct and incorrect attempts other than BKT and the design of the exercises could have a huge impact on the model (For example, the arrangements of easy and hard exercises).

3.5 Is the RNN representation meaningful?

Recurrent models have been successfully used in practical tasks like natural language processing [3]. These models can take days or even weeks to train. In a recently published paper, Wieting *et al.* [14] argue that RNNs might not be learning a meaningful state vector from the data. They show that a randomly initialized RNN model (with only W_o and b_o trained) can achieve similar results to models where all parameters are trained. This result is worrying because it may indicate that the RNN performance is due mostly to simply mapping input data to random high dimensional space. Once projected into the random vector space linear classification can perform well because points are more likely to be separated in a sparse vector space. The actual vector space may not be meaningful. We perform a similar experiment in training the DKT model. We randomly initialize the DKT model and only train the last linear layer (W_o and b_o) that maps the output of LSTM h_t to the skill vector, y_t . As shown in Table 1 and Table 2, the untrained recurrent network performs similarly to the trained network.

4. CONCLUSION AND FUTURE WORK

In this paper, we dive deep into the Deep Knowledge Tracing model. We have visualized and analyzed the behaviors of DKT through time using dimensionality reduction of the activations vector, a_t . We have also analyzed the temporal sequence behavior of DKT using qualitative and quantitative analyses. We find that the DKT model is most likely learning an ‘ability’ model, rather than tracking each individual skill. Moreover DKT is significantly impacted by the relative ordering of skills presented. We also discover that a randomly initialized DKT with only the final linear layer trained achieves similar results to the fully trained DKT model. In other words, the DKT model performance gains may stem from mapping input sequences into a random high dimensional vector space where linear classification is easier because the space is sparse. This is a worrying conclusion because it means the underlying recurrent representation may not be reliable nor semantically meaningful. Several mitigating measures are suggested in this paper, including the use of a loss function that mitigates unwanted behaviors and the use of an attention model to better capture long term skill dependencies. We leave evaluation of these suggestions to future work in the educational data mining community.

5. REFERENCES

- [1] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.
- [2] Corbett, A. T., and Anderson, J. R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4, 4, 253–278.
- [3] Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*
- [4] Feng, M., Heffernan, N. T., and Koedinger, K. R. 2006. Addressing the testing challenge with a web-based e-assessment system that tutors as it assesses. In *Proceedings of the 15th international conference on World Wide Web*, 307–316.
- [5] Graves, A., Wayne, G., and Danihelka, I. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.
- [6] Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9, 8, 1735–1780.
- [7] Karpathy, A., Johnson, J., and Fei-Fei, L. 2015. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*
- [8] Khajah, M., Lindsey, R. V., and Mozer, M. C. 2016. How deep is knowledge tracing? *arXiv preprint arXiv:1604.02416*
- [9] Krizhevsky, A., Sutskever, I., and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.
- [10] Maaten, L. V. D., and Hinton, G. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9, (Nov, 2008), 2579–2605.
- [11] Pavlik Jr, P. I., Cen, H., and Koedinger, K. R. 2009. Performance factors analysis—a new alternative to knowledge tracing. *Online Submission*.
- [12] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., and Sohl-Dickstein, J. 2015. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, 505–513.
- [13] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, 5998–6008.
- [14] Wieting, J., and Kiela, D. 2019. No training required: Exploring random encoders for sentence classification. *arXiv preprint arXiv:1901.10444*.
- [15] Xiong, X., Zhao, S., Van Inwegen, E. G., and Beck, J. E. 2016. Going deeper with deep knowledge tracing. In *International Educational Data Mining Society*.
- [16] Yeung, C. K., and Yeung, D. Y. 2018. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*.

Rank-Based Tensor Factorization for Student Performance Prediction

Thanh-Nam Doan
Department of Computer Science
University at Albany - SUNY
Albany, NY
tdoan@albany.edu

Shaghayegh Sahebi
Department of Computer Science
University at Albany - SUNY
Albany, NY
ssahebi@albany.edu

ABSTRACT

One of the essential problems, in educational data mining, is to predict students' performance on future learning materials, such as problems, assignments, and quizzes. Pioneer algorithms for predicting student performance mostly rely on two sources of information: students' past performance, and learning materials' domain knowledge model. The domain knowledge model, traditionally curated by domain experts, maps learning materials to concepts, topics, or knowledge components that are presented in them. However, creating a domain model by manually labeling the learning material can be a difficult and time-consuming task. In this paper, we propose a tensor factorization model for student performance prediction that does not rely on a predefined domain model. Our proposed algorithm models student knowledge as a soft membership of latent concepts. It also represents the knowledge acquisition process with an added rank-based constraint in the tensor factorization objective function. Our experiments show that the proposed model outperforms state-of-the-art algorithms in predicting student performance in two real-world datasets, and is robust to hyper-parameters.

Keywords

student modeling, predicting student performance, tensor factorization

1. INTRODUCTION

The popularity of online learning services and massive open online courses has led to extensive growth in the amount of student activity and learning data. As the number of students and learning materials increase in these online systems, the need for automatic sense-making from this data, educational data mining, becomes more evident. One of the important tasks in educational data mining is accurately predicting students' performance (PSP). PSP can be used in early detection of high-risk students that may fail or quit a class, in class evaluation and course planning activities,

and in learning material recommendation to students.

Many successful PSP techniques aim to predict students' performance in a problem by modeling their state of knowledge in different concepts required by that problem. To do this, pioneer and recent PSP techniques rely on the availability of a domain knowledge model that maps problems to concepts [19, 5, 25]. However, given the vast scope of learning materials in today's online learning systems, such domain knowledge models may not be available. Ideally, a PSP model should be able to work without requiring such a predefined map.

Additionally, a successful data mining model for PSP should be capable of considering specific characteristics of student learning process: (a) that students gain their knowledge on concepts over time, by practicing different problems, (b) that they may forget some of the gained knowledge, (c) that this knowledge gain is a gradual process, and (d) that learning can happen differently for different students in different problems and different times. Finally, to provide better insight to students and teachers, such a model should also be interpretable considering these characteristics. Previous research in the literature only cover some of the limitations above.

In this paper, we propose a student performance prediction model, Ranked-Based Tensor Factorization (RBTF), considering all the above requirements. To model student sequences on problems, we represent their scores over time as a three-dimensional tensor. To avoid the need for a predefined domain knowledge model, we propose a tensor factorization model for PSP, that maps problems and student knowledge in a lower-dimensional "latent" concept space. Representing student knowledge in this lower-dimensional space leads to a soft-membership approach that provides more flexibility by avoiding strict assignment of student knowledge to discrete "knowledge states". By learning student, problem, and time-based biases in this model we take into account the differences between students, problems, and times in the learning process. To capture the gradual learning requirement, we impose a rank-based constraint on student knowledge variables, that allows for occasional forgetting of concepts, but imposes a generally positive learning trend.

In our experiments, we study the proposed model in comparison with two state-of-the-art baseline PSP algorithms, on two real-world datasets. Our experiments show that our

Thanh-Nam Doan and Shaghayegh Sahebi "Rank-Based Tensor Factorization for Predicting Student Performance" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 288 - 293

model performs better than both baselines in the task of predicting student performance. We experiment with the performance and sensitivity of our model with various hyper-parameters.

Paper Outline. The remaining of the paper is organized as follows. Section 2 provides a brief literature review of the related work. Section 3 describes our model (RBTF) and the parameter learning steps. Section 4 evaluates extensively RBTF and other baselines on two real datasets. Lastly, Section 5 concludes the paper and suggests some directions for future works.

2. RELATED WORK

Many pioneer solutions to the problem of predicting student performance are based on either regression models [19] or Bayesian knowledge tracing (BKT) [5]. Regression-based models, such as performance factor analysis (PFA), try to predict students' performance using a pre-defined domain model that maps learning material to knowledge components [19]. PFA, which is based on learning factor analysis [4], takes into account prior successes and failures of a student on knowledge components associated with the current problem.

BKT is a constrained two-state hidden Markov model that models student knowledge in each knowledge component (KC) as two binary states: "known" and "unknown". It learns the probability of transitioning between these two states, and probabilities of students' success and failure in each KC, given their state of knowledge. Despite being successful in PSP for certain datasets, this model, in its original form, does not consider continuous states of knowledge or soft membership to knowledge states. Moreover, BKT does not capture the relationships between KCs, and is not personalized for individual student. Additionally, BKT also relies on a pre-defined domain model. Recently, new BKT-based models aim to address some of these problems [2, 9, 29]. For example, Pardos and Heffernan has addressed BKT's non-personalized modeling in [18, 17]. Song et al. proposed PSFK in [25] to address PSP when students encounter a knowledge component for the first time. But, these models rely on labeled problem knowledge components or concepts. Later, Gonzalez-Brenes and Mostow proposed a topical hidden Markov model that jointly learns the domain model and predicts student performance [10, 8]. However, this model has two restricting assumptions: that at each attempt, the student works on one skill of a problem, and that the students do not forget any acquired skills.

Recently, other approaches inspired by recommender systems' research and factorization models have been used for PSP. Despite being successful, these approaches are not tailored for the educational data mining problems specifically since they do not explicitly model student learning as a learning gain process. The matrix-factorization approaches in this area do not consider student sequences and only rely on a snapshot of student performance. For example, Thai-Nghe and Schmidt-Thieme proposed a multi-relational factorization student model that considers multiple relations between students and tasks, but does not consider student sequences [27]. Later, Nedungadi and Smruthy proposed a similar multi-relational matrix factorization approach ex-

ploring the effect of modeling biases [16]. Sahebi et al. also proposed another multi-relational learning approach that learned student performance according to canonical correlation analysis [22]. Non-negative matrix factorization has been used to improve performance predictions [28]. Pero et al. compared collaborative filtering techniques for the task of PSP in a small dataset [20]. Elbadrawy et al. predict student performance using their interactions with the learning management system to achieve a higher accuracy [7].

Some other recommender system-based approaches consider student sequence, but do not explicitly model knowledge gain in students. For example, Thai-Nghe et al. explored different factorization models, including tensor factorization, to predict student performance [26]. Sahebi et al. [23] studied educational data mining methods, such as PFA and BKT, with matrix and tensor factorization approaches, from the recommender systems literature, for PSP. Almutairi et al. have used tensor and coupled-matrix factorization to predict course-based student performance [1]. However, their tensor decomposition models do not explicitly model students' knowledge gain.

Although there have been some promising research on PSP that consider student sequence without requiring a domain model, these approaches have been limited. For example, SPARse Factor Analysis (SPARFA) by Lan et al. that uses Kalman filters to jointly learn the domain model, student knowledge, and the underlying question difficulties, can be very expensive to learn due to having a big state space [12]. Sahebi et al. have proposed a feedback-driven tensor factorization algorithm that can model student gradual knowledge acquisition [24]. But, their model has a strict constraint that does not allow for forgetting the concepts by students. Lindsey et al. proposed a non-parametric Bayesian technique that can refine the expert-labeled skills. However, they simplify the problem by finding coarse-grained skills as they restrict each problem to have exactly one skill [14]. In this paper, we propose a tensor factorization model for predicting student performance that does not require domain knowledge, models problems and student knowledge as soft-membership of latent concepts, and can model student sequences and gradual knowledge increase.

3. RANK-BASED TENSOR FACTORIZATION (RBTF)

Here we present our model, rank-based tensor factorization, by which we aim to predict students' performance in problems, considering their performance sequence and knowledge growth. Our proposed model is inspired by the recommender systems domain. Our choice of a recommender systems-based model was because of two main reasons: a) student performance similarities, and b) problem similarities. First, we consider that students with similar knowledge levels will perform similarly in solving problems. Second, we assume that a student will have similar performance on two problems with similar concepts. Recommender-based factorization models consider these two expectations. However, as discussed in the introduction section, a successful student model needs to include additional considerations. One of which is that knowledge gain is a gradual process for students, which happens over time. As students interact with learning materials, such as problems, they learn from them.

To represent this time-based process, we model students' activity sequences as a series of attempts on problems. For the student performance data to be represented according to these assumptions, we represent student sequences in a three-dimensional tensor (\mathcal{Y}), that has the student, problem, and time (attempt) dimensions. Each cell $y_{a,s,p}$ in this tensor represents student s 's score in problem p , that she has chosen to study at attempt a .

The core idea of the aforementioned assumptions is the notion of "concepts": gradual learning can be viewed as gaining knowledge in course concepts; student knowledge-based similarities are based on how much they mastered each of the concepts; and problem similarities are defined on how their represented concepts are shared. However, in many online educational systems, concepts are undefined and difficult to measure. In these systems, there are no "observed" features defined as problem concepts or knowledge components. Hence, we propose to discover shared "latent" features between students and problems as representatives for the notion of concepts. We model each problem as a vector of k latent concepts, that shows the importance of each latent concept in that problem. Also, we model each student's knowledge at any time point as another vector of the same latent concepts.

We assume that a student s 's performance on problem p at time a is a result of her existent knowledge in the latent concepts required by the problem. Accordingly, we model estimated student score $\hat{y}_{a,s,p}$ as a dot product between problem's latent concept vector \mathbf{q}_p and student's knowledge in those concepts $\mathbf{t}_{a,s}$:

$$\hat{y}_{a,s,p} \approx \mathbf{t}_{a,s} \cdot \mathbf{q}_p \quad (1)$$

To maintain the interpretability of our model, we enforce latent variables in \mathbf{q}_p to be non-negative. Here, by choosing the number of concepts (k) less than the number of problems and students, we are representing students and problems in a lower-dimensional latent space that can better capture student and problem similarities (our second and third assumptions). However, the model in Equation 1 does not consider differences in factors such as student ability, problem difficulty, or student cohort strength. For example, students' average score in a more difficult problem is expected to be less than their average score in an easier problem. To address this issue, we add student, problem, and attempt biases (b_s , b_p , and b_a), in addition to an overall cohort bias (μ) to our model:

$$\hat{y}_{a,s,p} \approx \mathbf{t}_{a,s} \cdot \mathbf{q}_p + b_s + b_p + b_a + \mu \quad (2)$$

To learn the parameters of this problem (\mathcal{T} , Q , b_s , b_p , b_a , and μ) we minimize the objective function in Equation 3. The first component calculates the squared difference between observed student scores and estimated student scores. The last three components are for regularizing biases, student knowledge, and problem concepts for generalizability purposes.

$$\begin{aligned} \mathcal{L}_1 = \sum_{a,s,p} (\hat{y}_{a,s,p} - y_{a,s,p})^2 \\ + \lambda(b_s^2 + b_p^2 + b_a^2) + \lambda_1 \|\mathbf{t}_{a,s}\|^2 + \lambda_2 \|\mathbf{q}_p\|^2 \end{aligned} \quad (3)$$

The model in Equation 2 does not address our gradual learning assumptions for students. To capture this gradual learning, we can assume that a student's knowledge ($\mathbf{t}_{a,s}$) increases over time. But, we should also note that this knowledge increase depends on the problems that the student selects to solve and the concepts presented in them. As a result, we can translate this knowledge increase as an increase in estimated student scores in problems ($\mathbf{t}_{a,s} \cdot \mathbf{q}_p$). In other words, we expect that student s 's predicted scores at attempt a to be larger than her scores at attempt $a-1$:

$$\mathbf{t}_{a,s} \cdot \mathbf{q}_p - \mathbf{t}_{a-1,s} \cdot \mathbf{q}_p \geq 0$$

In reality, this knowledge increase can be non-monotonic. For example, a student may forget some concepts after a while. For this reason, we propose to use a rank-based model for student knowledge gain, that allows knowledge loss to happen for students, but penalizes it. Using this rank-based model, we aim to maximize the difference between the aggregation of all students' scores on all questions at each attempt versus the attempts before that. Hence, we would like to maximize \mathcal{L}_2 in Equation 4. Here, $\sigma(\cdot)$ is the sigmoid function, defined as $\sigma(x) = 1/(1 + e^{-x})$. Sigmoid function is selected because of its superiority in rank-based recommendation systems [21, 6]. The term $\log(\sigma(\mathbf{t}_{a,s} \cdot \mathbf{q}_p - \mathbf{t}_{j,s} \cdot \mathbf{q}_p))$ means that for attempt a of student s , the *ranking* of s 's score at a is higher than the one of s at j with $j < a$.

$$\mathcal{L}_2 = \sum_{j=1}^a \sum_s \sum_p \log(\sigma(\mathbf{t}_{a,s} \cdot \mathbf{q}_p - \mathbf{t}_{j,s} \cdot \mathbf{q}_p)) \quad (4)$$

To capture the dynamics between all assumptions, we combine the minimization of \mathcal{L}_1 in Equation 3 and maximization of \mathcal{L}_2 in Equation 4. Our final objective is to minimize the loss function in Equation 5. The hyper-parameter ω is to control the relative strictness of knowledge increase versus the importance of having a more accurate estimate of student performance.

$$\begin{aligned} \mathcal{L} = \sum_{a,s,p} (\hat{y}_{a,s,p} - y_{a,s,p})^2 + \lambda_1 \|\mathbf{t}_{a,s}\|^2 + \lambda_2 \|\mathbf{q}_p\|^2 \\ + \lambda(b_s^2 + b_p^2 + b_a^2) - \omega \sum_{j=1}^a \sum_s \sum_p \log(\sigma(\mathbf{t}_{a,s} \cdot \mathbf{q}_p - \mathbf{t}_{j,s} \cdot \mathbf{q}_p)) \end{aligned} \quad (5)$$

Learning the Parameters: By using stochastic gradient descent algorithm to minimize \mathcal{L} , we find student knowledge of each latent concept at any point, the importance of each latent concept in each problem, and estimation of student score in each problem at any attempt. Recall that the parameters we want to infer are \mathcal{T} , Q , b_s , b_p , b_a ,

and μ . For the cohort bias μ , we assign the average score of all students on all problems [11], i.e. $\mu = \frac{\sum_{a,s,p} y_{a,s,p}}{\sum_{a,s,p} \mathcal{I}(a,s,p)}$ where $\mathcal{I}(a,s,p)$ is an indicator function returning 1 if the tuple (a,s,p) is in our training set; otherwise 0.

4. EXPERIMENTS

In the following, we evaluate our model in comparison with two state-of-the-art methods in the task of predicting student performance. Further, we analyze how our solution models students' learning process by looking at students' knowledge gain in course concepts. Eventually, we experiment on RBTF's sensitivity to various hyper-parameter settings.

4.1 Dataset and Experiment Setup

For experiments, we use the Canvas network dataset¹ which is available online [3]. Canvas Network hosts many freely available open online courses. In addition to learning modules, each course can have different types of assignments, discussions, and quizzes. In this platform, participants are not limited to a specific sequence of learning material or assignments. The dataset is anonymized such that student IDs, course names, discussion contents, submission contents, or course contents are not available.

Dataset	#students	#problems	#attempts	Avg. attempts
Course 1	531	91	87	29.92
Course 2	2597	32	30	12.73

Table 1: Dataset Statistic.

We select two courses in Canvas and denote them as Course 1 and Course 2. The selected courses have the most number of quizzes in the whole dataset. We consider each quiz as a problem in our model. Quizzes are graded between zero and a maximum possible score. For consistency, we normalize the quiz grades between zero and one. Table 1 shows the statistics of these two courses. As shown in the table, Course 2 has more students but less number of problems and attempts than Course 1.

The data of each course is represented as a list of tuples (*attempt, student id, quiz id, grade*). We randomly split 80% of tuples for training and the remaining (i.e. 20%) for testing.

Hyper-parameter Setting: In the performance prediction experiments (Section 4.2), we set $\omega = 0.5$, $\lambda_1 = \lambda_2 = 0.1$ and regularization of bias $\lambda = 0.001$. The number of concepts is set to 3.

4.2 Student Performance Prediction

In this section, we compare the prediction performance of RBTF with other baselines to evaluate the prediction ability of RBTF.

Baselines: To compare the prediction performance, we employ the following two baselines:

¹<http://canvas.net>

- **Feedback-Driven Tensor Factorization (FDTF)** [24]: It is a tensor factorization model specifically tailored to predict students' performance. It considers students' gradual learning process. However, the assumption of hard constraint on knowledge increase in students limits its modeling capacity. Also, it does not include biases and does not allow for the concepts to be forgotten by students.
- **SPARse Factor Analysis (SPARFA)** [13]: SPARFA is a probabilistic factor analysis approach that calculates the probability of a student's correct response to a problem. It does not require a predefined domain knowledge model. However, it does not consider students' sequences. To adapt it to our problem, we use the probability scores instead of the predicted student grade.

Metrics: We use two measures to evaluate the performance prediction task. Since our main goal is to predict student scores or grades, we would like to measure how close our predictions are to the actual student scores. To do this, we use the root mean squared error (RMSE). The lower the value of RMSE, the better the model.

Since many performance prediction models focus on predicting students' success and failure as a binary value, instead of their score [13, 5], we also employ accuracy for performance comparison. To do this, we regard scores greater than 0.5 as success and the rest as failure. Unlike RMSE, the higher the value of accuracy, the better the model.

Dataset	RMSE			Accuracy		
	RBTF	FDTF	SPARFA	RBTF	FDTF	SPARFA
Course 1	0.12	0.27	0.59	92.5%	85.2%	81.7%
Course 2	0.2056	0.2116	0.567	95.24%	92.8%	87.41%

Table 2: Prediction Performance.

Results: Table 2 shows the prediction performance of our model (RBTF) and the two baselines (FDTF and SPARFA) on the two datasets. As we can see, both tensor factorization models (RBTF and FDTF) perform better than SPARFA in both courses. This shows the importance of considering student sequences in predicting their performance. Also, we can see that RBTF performs better than FDTF in both courses. This shows that, even though modeling sequential knowledge increase in students is important, this increase should not be strictly monotonic and should be flexible to allow for occasional forgetting of concepts.

4.3 Hyper-parameter Sensitivity Analysis

In this section, we study RBTF's sensitivity to hyper parameter values. First, we experience on the balance between training error on student performance fitting (\mathcal{L}_1 in Equation 3) versus modeling student knowledge increase (\mathcal{L}_2 in Equation 4) on the generalizability of our model. To do this, we measure the test error by varying hyper-parameter ω , that controls this balance in Equation 5. Then, we capture the effect of the number of concepts on RBTF's performance by varying k in Equation 5 and measuring its error on test data.

Dataset	0.01	0.25	ω 0.5	0.75	1.0
Course 1	0.191	0.128	0.12	0.137	0.141
Course 2	0.233	0.2064	0.2056	0.2154	0.2224

Table 3: RMSE with different value of ω and number of concept is 3.

Sensitivity to ω : Recall that ω controls the trade-off between having an accurate estimation of student performance and the constraint of knowledge increase. A larger value of ω , imposes more contribution of knowledge increase constraint to the performance of RBTF, and a smaller value of ω dictates a stricter fit of student performance to the training data. We use different values of ω from 0 to 1 and measure RBTF’s RMSE corresponding to these values. For other parameters, we use the default values mentioned in Section 4.1. Table 3 presents the performance of RBTF with different values of ω on the two datasets. From the table, we observe that $\omega = 0.5$ yields the best performance of RBTF and it is consistent for the two datasets. However, the results from Course 2 dataset is more sensitive to the changes in ω . One reason for this can be the smaller number of attempts and more sparsity of Course 2 dataset, compared to Course 1 dataset, that can lead to easier overfitting to training data.

Dataset	3	k 5	10	15
Course 1	0.12	0.122	0.127	0.128
Course 2	0.2056	0.206	0.2065	0.2065

Table 4: RMSE with different value of number of concepts and $\omega = 0.5$.

Sensitivity to k : Recall that, in our model, concepts are latent lower-dimensional representations of student performance and problems over attempts. They can be used to model the similarity between students and problems. To measure the effect of the number of concepts k , we tune the value of k while using the default values for other parameters (see Section 4.1). We measure the RMSE of RBTF by changing k . Table 4 shows the results. From the table, we observe that increasing the value of k makes RBTF perform slightly worse. This finding is consistent in both datasets. However, RBTF is relatively robust to k as this increase in error is minor.

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel rank-based tensor factorization method (RBTF), which is able to predict the performance score of students by considering the gradual learning of students as a ranking problem. Our model has the flexibility to present student knowledge as a soft-membership of latent concepts, only requires activity sequences of students, and discovers individualized student knowledge model including biases. Our extensive evaluations show that RBTF outperforms state-of-the-art baselines in both root mean square error and accuracy measures. Also, we show our models robustness to hyper-parameters by experimenting the balance between knowledge ranking and performance fitting parts of the model, and by varying the number of latent concepts.

There are several directions to extend this research work

further. In this work, we experiment on performance prediction within the same course. This model can be used to experiment on between-course performance predictions. Another application of our model is to detect knowledge gaps in students and recommend useful learning materials to them. Moreover, contingent on the availability of a domain knowledge model, this work can be extended to improve the existing domain knowledge model. Recent studies show that order and length of students’ activities are essential for understanding students’ performance [15]. So, integrating these features can enhance the prediction performance of our model.

Acknowledgement. This material is based upon work partially supported by the National Science Foundation under Grant No. 1755910.

6. REFERENCES

- [1] F. M. Almutairi, N. D. Sidiropoulos, and G. Karypis. Context-aware recommendation-based learning analytics using tensor and coupled matrix factorization. *IEEE Journal of Selected Topics in Signal Processing*, 11(5):729–741, 2017.
- [2] R. S. Baker, A. T. Corbett, and V. Aleven. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *International conference on intelligent tutoring systems*, pages 406–415. Springer, 2008.
- [3] Canvas-Network. Canvas network courses, activities, and users (4/2014 - 9/2015) restricted dataset, 2016.
- [4] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis—a general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems*, pages 164–175. Springer, 2006.
- [5] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [6] T.-N. Doan and E.-P. Lim. Modeling location-based social network data with area attraction and neighborhood competition. *Data Mining and Knowledge Discovery*, 33(1):58–95, 2019.
- [7] A. Elbadrawy, R. S. Studham, and G. Karypis. Collaborative multi-regression models for predicting students’ performance in course activities. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 103–107. ACM, 2015.
- [8] J. González-Brenes. Modeling skill acquisition over time with sequence and topic modeling. In *Artificial Intelligence and Statistics*, pages 296–305, 2015.
- [9] J. González-Brenes, Y. Huang, and P. Brusilovsky. General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge. In *The 7th International Conference on Educational Data Mining*, pages 84–91. University of Pittsburgh, 2014.
- [10] J. Gonzalez-Brenes and J. Mostow. What and when do students learn? fully data-driven joint estimation of cognitive and student models. In *Educational Data Mining 2013*, 2013.

- [11] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.
- [12] A. S. Lan, C. Studer, and R. G. Baraniuk. Time-varying learning and content analytics via sparse factor analysis. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 452–461. ACM, 2014.
- [13] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *The Journal of Machine Learning Research*, 15(1):1959–2008, 2014.
- [14] R. V. Lindsey, M. Khajah, and M. C. Mozer. Automatic discovery of cognitive skills to improve the prediction of student learning. In *Advances in neural information processing systems*, pages 1386–1394, 2014.
- [15] M. Mirzaei, S. Sahebi, and P. Brusilovsky. Annotated examples and parameterized exercises: Analyzing students’ behavior patterns. In *International conference on artificial intelligence in education*, 2019.
- [16] P. Nedungadi and T. Smruthy. Personalized multi-relational matrix factorization model for predicting student performance. In *Intelligent Systems Technologies and Applications*, pages 163–172. Springer, 2016.
- [17] Z. A. Pardos and N. T. Heffernan. Modeling individualization in a bayesian networks implementation of knowledge tracing. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 255–266. Springer, 2010.
- [18] Z. A. Pardos and N. T. Heffernan. Using hmms and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. *Journal of Machine Learning Research W & CP*, 2010.
- [19] P. I. Pavlik Jr, H. Cen, and K. R. Koedinger. Performance factors analysis—a new alternative to knowledge tracing. *Online Submission*, 2009.
- [20] Š. Pero and T. Horváth. Comparison of collaborative-filtering techniques for small-scale student performance prediction task. In *Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering*, pages 111–116. Springer, 2015.
- [21] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 452–461. AUAI Press, 2009.
- [22] S. Sahebi and P. Brusilovsky. Student performance prediction by discovering inter-activity relations. *Educational Data Mining*, pages 87–96, 2018.
- [23] S. Sahebi, Y. Huang, and P. Brusilovsky. Predicting student performance in solving parameterized exercises. In *Intelligent Tutoring Systems*, pages 496–503. Springer, 2014.
- [24] S. Sahebi, Y. Lin, and P. Brusilovsky. Tensor factorization for student modeling and performance prediction in unstructured domain. In *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016, Raleigh, North Carolina, USA, June 29 - July 2, 2016*, pages 502–506, 2016.
- [25] Y. Song, Y. Jin, X. Zheng, H. Han, Y. Zhong, and X. Zhao. Psfk: a student performance prediction scheme for first-encounter knowledge in its. In *International Conference on Knowledge Science, Engineering and Management*, pages 639–650. Springer, 2015.
- [26] N. Thai-Nghe, T. Horváth, and L. Schmidt-Thieme. Factorization models for forecasting student performance. In *Educational Data Mining 2011*, 2010.
- [27] N. Thai-Nghe and L. Schmidt-Thieme. Multi-relational factorization models for student modeling in intelligent tutoring systems. In *2015 Seventh International Conference on Knowledge and Systems Engineering (KSE)*, pages 61–66. IEEE, 2015.
- [28] K. Xu, R. Liu, Y. Sun, K. Zou, Y. Huang, and X. Zhang. Improve the prediction of student performance with hint’s assistance based on an efficient non-negative factorization. *IEICE TRANSACTIONS on Information and Systems*, 100(4):768–775, 2017.
- [29] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*, pages 171–180. Springer, 2013.

Using Latent Variable Models to Observe Academic Pathways

Nate Gruver
Stanford University
ngruver@cs.stanford.edu

Ali Malik
Stanford University
malikali@cs.stanford.edu

Brahm Capoor
Stanford University
brahm@cs.stanford.edu

Chris Piech
Stanford University
piech@cs.stanford.edu

Mitchell L. Stevens
Stanford University
stevens4@stanford.edu

Andreas Paepcke
Stanford University
paepcke@cs.stanford.edu

ABSTRACT

Understanding large-scale patterns in student course enrollment is a problem of great interest to university administrators and educational researchers. Yet important decisions are often made without a good quantitative framework of the process underlying student choices. We propose a probabilistic approach to modelling course enrollment decisions, drawing inspiration from multilabel classification and mixture models. We use ten years of anonymized student transcripts from a large university to construct a Gaussian latent variable model that learns the joint distribution over course enrollments. The models allow for a diverse set of inference queries and robustness to data sparsity. We demonstrate the efficacy of this approach in comparison to others, including deep learning architectures, and demonstrate its ability to infer the underlying student interests that guide enrollment decisions.

1. INTRODUCTION

Education researchers increasingly recognize the need to understand the sequential accumulation of college coursework into academic pathways. In [2, 23], Bailey et al. call for change in how colleges organize course offerings to enable more efficient pathways. Rather than presenting a bewildering array of courses, cafeteria-style, they recommend “guided pathways” through academic offerings. Baker [3] builds on Bailey’s work, suggesting “meta-majors” for simplifying choice without curtailment of options. Meta-majors entail combining coursework supporting multiple majors into larger, substantively coherent content domains. Baker proposes social-network analytic techniques to discover opportunities for building meta-majors. All of these authors argue that rather than limiting choice, such interventions can yield more tractable programs, faster degree completion, and lower cost for both students and schools.

Such reforms can be enabled by analysis of data corpora

describing the academic sequences of prior student enrollments. For example, some courses may be de facto prerequisites for other courses, whether listed as “required” or not in formal catalogue entries. Similarly, “odd” delays in taking particular courses, or unexplained detours in course selection, can be symptoms of unintended scheduling conflicts.

In the service of such reforms, we offer a model of course enrollment capable of efficient inference over hundreds to thousands of classes. Our generative model captures the full joint distribution of course enrollments and can be used to sample potential pathways for any given student. The model’s complexity allows us to determine an underlying “typography” of students, from implicit course-taking patterns to differing levels of novelty in their academic pathways relative to the overall population of paths.

2. BACKGROUND & MODELS

Predicting course enrollment decisions may be viewed as a problem of multi-label classification: the task of assigning a subset of labels to each data point in a collection. In context of academic course enrollments, each data point is a student and the labels are courses enrolled. The problem of modeling all possible enrollment choices scales exponentially with the number classes ($O(2^N)$), which motivates a statistical approach. Probabilistic graphical models (PGMs) and deep neural networks are perhaps the most prominent methods for stochastic models of high-dimensional data. As our motivation in this work is not simply high accuracy but also interpretability and inference, we focus on PGMs, which fare better on those aspects and are amenable to scaling adequate for our empirical setting.

2.1 Latent Variable Models

Latent variable models are a subclass of PGMs in which some variables are never observed in training data and are thus “latent.” These models are more computationally demanding than fully observed models, but also are able to capture complex structure in data without supervision.

2.1.1 Models of Conditional Independence

Among the simplest and most commonly used latent variable models is the naive Bayes model with hidden variable H taking discrete values h_i and observations X . In the enrollment setting, $X = [x^0, \dots, x^N]$ and $x^i = [x_0^i, \dots, x_M^i]$ with

Nate Gruver, Ali Malik, Brahm Capoor, Chris Piech, Mitchell Stevens and Andreas Paepcke “Using Latent Variable Models to Observe Academic Pathways” In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 294 - 299

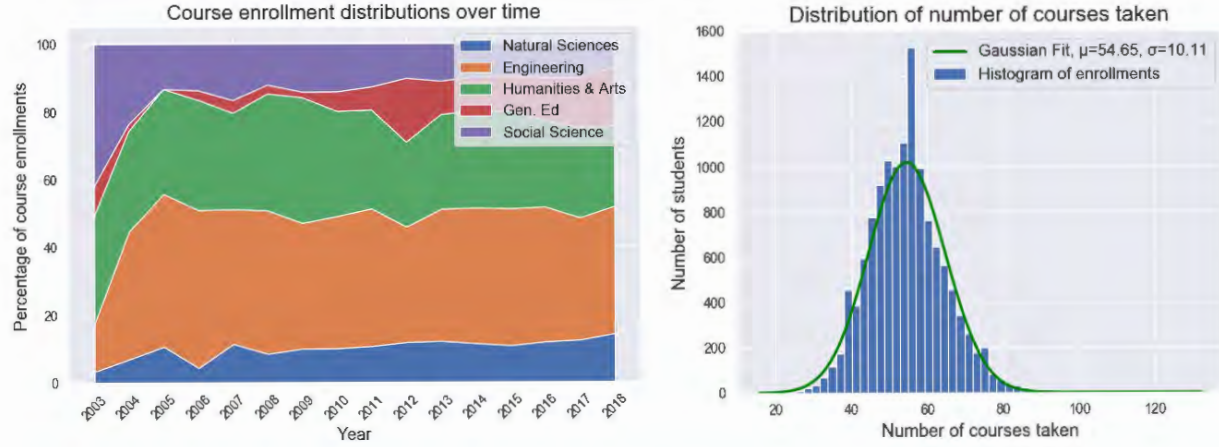


Figure 1: Top: A stacked area plot of enrolled courses per university sub-school per year. Bottom: A histogram of the number of courses taken by each individual student with a Gaussian fitting.

$x_j^i \in \{0, 1\}$ (0 denoting no enrollment and 1 an enrollment). The generative process of the model is described below:

$$\begin{aligned} h^i &\sim \text{Multinomial}(\theta) \\ x_j^i | h^i &\sim \text{Multinomial}(\phi_j) \end{aligned}$$

Given the value of the hidden variable, each individual probability of enrolling in a course is independent. This is a poor inductive bias because enrollment decisions often influence one another, and the number of courses taken by a student in a given year is dependent on the courses taken. It is easier to capture these two facets of the data if we can model enrollments jointly, without strong independence assumptions.

2.1.2 Gaussian Mixture Model

Any joint probability distribution over all discrete combinations of $x^i \in \{0, 1\}^m$ requires $2^m - 1$ parameters and is thus intractable. One possible solution is relaxation of the discrete problem to a real-valued vector space with $\bar{X}^i \in \mathbb{R}^m$ and

$$x_j^i = \begin{cases} 1 & \bar{x}_j^i > 0 \\ 0 & \text{else} \end{cases}$$

By training a model over \bar{X} , we can take advantage of real-valued distributions with much smaller parameter spaces.

The Gaussian Mixture Model (GMM) is an archetypal latent variable model for real-valued data [22]. We can describe a GMM by generative process below:

$$\begin{aligned} h^i &\sim \text{Multinomial}(\theta) \\ \bar{x}^i | h^i &\sim \mathcal{N}(\mu, \Sigma) \end{aligned}$$

We can modify the GMM for the setting of multi-label classification by providing an unbiased estimator of the probability of each binary sample:

$$\begin{aligned} P(x = [1, 0, \dots, 1]) &= P(\bar{x}_0 > 0, \bar{x}_1 < 0, \dots, \bar{x}_0 > 0) \\ &\approx \frac{1}{K} \sum_{i=1}^K \Phi(\vec{0}; \mu(y^i), \Sigma(y^i)) \mathbb{1}[y^i > \vec{0}] \end{aligned}$$

where y^i are samples drawn from a multivariate normal over a subset of the variables in x and $\mu(y^i)$, $\Sigma(y^i)$ are the parameters of a multivariate normal conditioned on the value of y^i . More detail on this estimator is provided in the online posting of this paper.

At face value, it might seem odd to model enrollments as Gaussian-distributed. We choose this particular model both because it makes our real-valued relaxation tractable and because we think it is reasonable to assume enrollments within each cluster will be fairly unimodal and smooth, especially as sample sizes increase.

2.1.3 Contextual Mixture Model

Hidden Markov Models (HMMs) are a common extension of stationary mixture models to sequential data [21]. In these models, the single latent variable is replaced with a Markov chain of hidden states. This model is naturally recursive, a property that is extremely useful when modeling processes that are positive recurrent. However, as enrollments often exhibit a strict order and returning to previous states is unlikely, we prefer a model that is strictly time-dependent or, as we will call it here, “contextual.” In general any Contextual Mixture Model (CMM) can be expressed using a Hidden Markov model, but enforcing this structure allows us to incorporate priors that significantly improve the chances of training a plausible model.

For a CMM with Gaussian emission probabilities, we have

$$\begin{aligned} h^0 &\sim \text{Multinomial}(\theta) \\ h^{t+1} | h^t &\sim \text{Multinomial}(\phi^t) \\ \bar{x}^t | h^t &\sim \mathcal{N}(\mu_i^t, \Sigma_i^t) \end{aligned}$$

Note that the parameters of the transition and emission distributions are different for each timestep. Figure 2 shows a diagram of our proposed model in plate notation.

The small, discrete latent space of our model offers highly interpretable representations compared with the continuous latent vector space of neural architectures (see Fig. 4) and inference is highly efficient as the model has low tree-width.

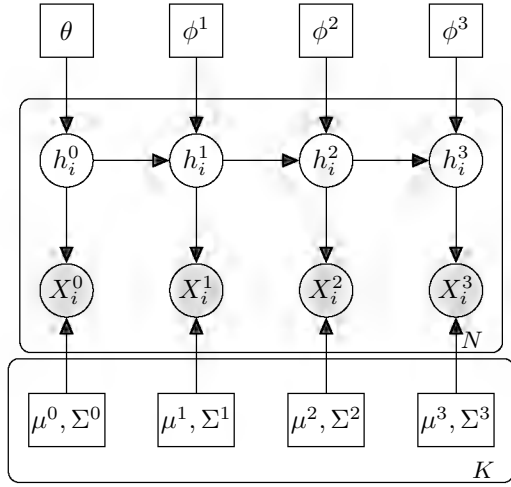


Figure 2: Graphical representation of contextual mixture model using plate notation

Our discrete probability estimator also allows modeling of all courses jointly, which is essential in this setting.

2.1.4 Parameter Learning

There are two primary methods of learning the parameters of the model we propose: expectation-maximization (EM) [8, 5] and gradient descent [6] on the log-likelihood objective. The approximate probability estimate of the model also creates the possibility for differing levels of precision in accordance with the amount of computation one is willing to invest. For a highly biased learning process, one can simply train the model on data shifted from $[0, 1]$ to $[-1, 1]$ using the exact probability estimate. For a less biased learning process, one can use the probability estimator described in Section 2.1.2 and unbiased estimator of the gradient with respect to the model parameters. For more details, see the online posting of the paper.

2.2 Baseline Models

In Section 4 we present a comparison of our model with three baseline models. The first of these is a naive Bayes model with the strongest model assumptions. The second is tree-augmented naive Bayes [9], which adds dependencies between variables to better model the joint density. Both models with trained with EM.

The last model we use for comparison is a Variational Autoencoder (VAE)—a deep generative model [15]. We use a simple VAE with two fully connected layers in the encoder and decoder, trained on binary cross-entropy loss.

It is important to note that while the VAE offers a good comparison point, the type of conditional inference (over sets of courses) that we describe for our Gaussian relaxation are not tractable in a standard VAE framework. In fact, VAE models can suffer from suboptimal inference in general when there is overfitting of the decoder network [7]. This issue is particularly concerning in this setting with relatively small sample size.

3. EXPERIMENTS

In this section, we describe the data used to train the model presented in Section 2 and how we evaluated them during training.

3.1 Data

We use eighteen years of course enrollment data from a large private university in the United States. The data comprise approximately 30,000 student enrollment records with fields for course name and student major. We removed part-time and summer students from the dataset, limiting the analyses presented here to full-time academic-year enrollments only.

Figure 1 shows two basic visualizations of the data after pre-processing. There are at least two notable takeaways from these plots. First, the proportion of enrollments in each academic division within the university remains relatively stable through most of the time period represented in the dataset. We use this fact to aggregate over time without explicitly modeling changes in enrollment patterns. Second, the fact that the number of courses taken is approximately Gaussian-distributed shows that enrollment patterns are not intensely multi-modal; thus the assumptions of probabilistic model are plausible.

In what follows we replace full course names with abbreviated proxies to enable universal legibility. For example, CS1 corresponds to the introductory computer science class and “Alg” or “AI” correspond to algorithms or artificial intelligence classes respectively.

4. EVALUATION

4.1 Mean-Field Evaluation

Though we can compare many of the models under consideration with log-likelihood alone, some only offer an approximate lower bound (VAEs). Thus we provide another evaluation metric that can be used to compare any model that can generate sampled enrollments.

For this loss function, we compare the empirical enrollment distributions in samples from our model and the distributions of the hold-outs. Let p_j^t be the probability that class j is taken by any given student in the hold-out data, and p_j^s be the corresponding probability in the samples. We take as our error, $E(p^t, p^s)$ with

$$E(p^t, p^s) = \sum_j (p_j^t - p_j^s)^2$$

which approximates the distance between the two true multivariate distributions—the distribution of our model and the distribution of the data—if all the variables were independent (mean field approximation).

4.2 Sample Quality

In Figure 3 we compare the performance of our model on *hold-out data* relative to baseline models described in Section 2.2. We also include a direct comparison of the best performance for each model in Table 1.

In Figure 3 we can see that our proposed model outperforms the two baselines across the board. It also is evident that the VAE baseline suffers bad generalization as the complexity of

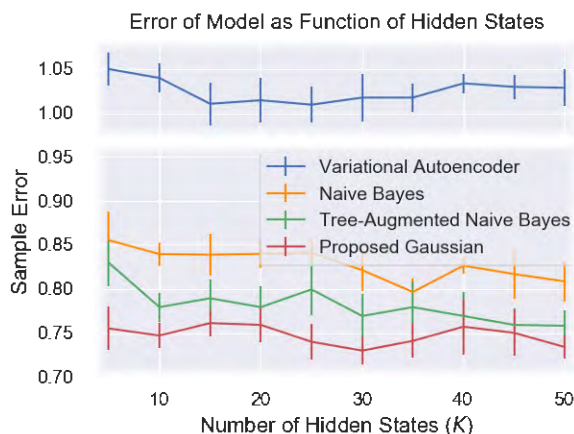


Figure 3: Plots of the error for the proposed model and other models. The parameter K is the dimension of the latent space.

Model	Sample Error	Inference Acc.
Deep Generative	1.01	N / A
Naive Bayes	0.78	60%
Tree-Augment Naive Bayes	0.76	66%
Our Gaussian	0.72	86%

Table 1: A direct comparison of the best performance from each model on hold-out data

the model increases. These models were trained adaptively, according to performance on the validation set, and thus are not simply underfitting due to increased training complexity.

In comparing the graphical models, increased complexity—both in the observation model and latent space—leads to lower error. As the error calculation itself makes an independence assumption, it is not surprising that the performance of all three graphical models is relatively close. The true dominance of the model proposed here is perhaps most evident in the inference task of Table 1, described in Section 5.3.

4.3 Visualizing Hidden Variables

Beyond using the loss function defined in Section 4.1, we can also examine the hidden states of a trained model to validate the learning process. In particular, we can investigate whether the hidden space captures semantically meaningful categories. Figure 4 shows a visualization for our model trained on CS majors. The clusters in the grid correspond to required courses for three different concentration within the major¹, and the color shows the most likely latent state assigned by the model to each course. As we can see, courses within the same concentration are assigned strikingly similar latent states by the model, suggesting that the model captures a semantically meaningful notion of the different concentrations in its hidden state. Therefore, if there are unknown correlations in course enrollments—for example

¹These requirements were taken from the department website: <https://exploreddegrees.stanford.edu/schoolofengineering/computerscience>.

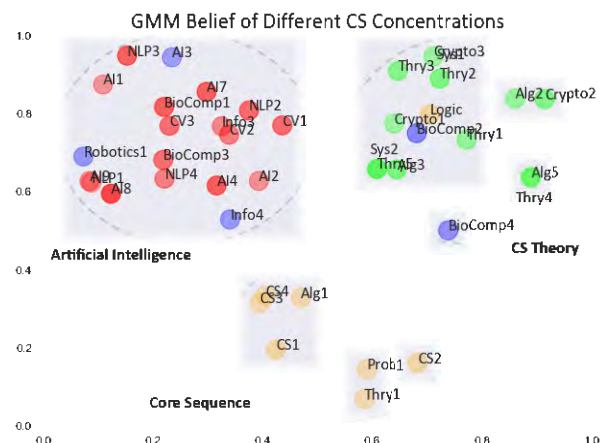


Figure 4: A visualization of the semantic meaning captured by the latent space of the model. Coloring corresponds to hidden state, and translucency indicates the confidence of the model.

many AI and biology courses taken together—this model could bring these patterns to the fore, allowing administrators an insight into possible ways to improve their degree concentrations.

5. APPLICATIONS

In this section we present results from two different experiments performed with our proposed model. These applications demonstrate only a fraction of the model’s scope, but show its power to provide insights.

5.1 Quantifying Enrollment Likelihood

One of the useful applications made simple by our generative model is in quantifying enrollment likelihood. A model trained on student enrollments will approximate the distribution of the training data. Thus if we evaluate the likelihood of a new student’s enrollments given the model, we can get a sense of how this student differs from the training examples. Taking this principle to its extreme, we can train a model for each student on every other student’s enrollments, allowing us to model exactly how much each particular student varies from the typical.

By examining the classes taken by students who are evaluated as high versus low likelihood, we see that the model captures at least two meaningful axes of variance. Firstly, it recognizes that it is rare for students to take a very diverse set of courses spanning many academic subjects. This insight is demonstrated in Figure 6, which shows the average coursework for each type of student. The second insight that the model captures is the spectrum of ambition. More specifically, the model places very low probability on the small subgroup of students that take up to 30 computer science classes and places high probability on taking just the core requirements of the degree². Atypical students take about 20 more courses than their counterparts on average.

²We can identify this trend by looking at the exact classes that are most commonly taken by these students e.g. the core requirements.

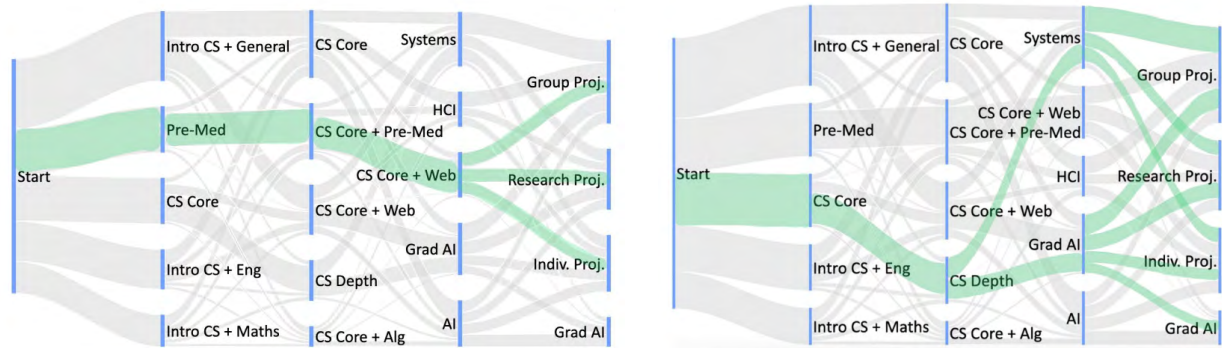


Figure 5: Two shadings of the same Sankey diagram constructed from the CMM trained on CS undergraduate enrollments. Top: A common path taken by students engaging in pre-med requirements is highlighted in blue. Bottom: A common path for students committed to in-depth study of computer science is highlighted.

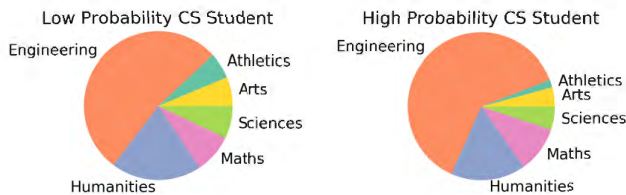


Figure 6: Pie charts representing the difference between enrollment patterns captured by the model. Sections correspond to the average number of courses taken in a academic subject.

5.2 Understanding Pathways

Another capability of the model presented here is the ability to analyze sequences of enrollments, from inferring likely paths between X and Y, to uncovering unspoken student strategies. We can display this visually using Sankey diagrams in which the width of the line between adjacent segments is proportional to the transition probability between the corresponding hidden states in the model. Figure 5 shows this style of Sankey for CS students. In this diagram, we can note the two types of paths highlighted in the diagram. The first of these captures students who were actively taking the pre-medical requirements freshman and sophomore year. These same students were subsequently much more likely to take depth courses later and were more likely to focus on web development of information systems in their depth courses. We can contrast these students with the students that are highly committed to the CS major and its core classes starting freshman year. These students are much more likely to enroll in depth classes by their sophomore year and are predisposed towards the systems and AI concentrations within the major.

5.3 Inferring Intermediate Classes

Another unique capability of our model is inferring the likelihood of intermediate classes. Given the classes taken freshman year and goal classes for senior year, the model can place a likelihood on intermediate classes. One possible use case for this ability is inference of soft or tacit prerequisites for courses.

To test this aspect of the model, we predicted whether students would take each of 5 common classes in their sophomore year given freshman and senior year enrollments. We were able to recover the correct enrollment with around 86% probability. From this result it is clear that the model can learn a sensible joint distribution over multi-year enrollments. We can compare this performance with that of the baseline models in Table 1, noting a substantial gain.

An even more interesting use case of this inference ability, however, is not simply prediction of common courses, but the potential for improving course selection tools. Only a model that captures the temporal dependencies across all courses is capable of offering helpful insights for goal-directed course selection.

6. PRIOR WORK

Much of the prior work on enrollment modeling in the university setting is dedicated purely to predictive models of future course enrollment [13, 18, 24] and academic performance [16, 11]. These models are largely incapable of producing the kinds of insights shown here. Preliminary work has also seen application of clustering algorithms to enrollments in form of latent variable models like Latent Dirichlet Allocation (LDA) [17] and recurrent neural networks [20].

Much of the state-of-the-art research in student decision modeling is now found in the study of massive open online courses (MOOCs). Gardner and Brooks [10] provide a thorough overview of modern models for the problem setting. Of note, Balakrishnan and Coetzee use a Hidden Markov Model (HMM) to predict attrition in MOOCs [4]. Similarly, Al-Shabandar et al. use Gaussian Mixture Models (GMMs) to cluster MOOC students at each timestep, and thus identify clusters of students that are likely to withdraw from the courses [1]. Both of these models resemble ours though their task is prediction of simple binary outcomes.

Work in course recommender systems is also inspiring. Khorasani et al. create a recommender based on a Markov model [14]. Jiang et al. use a neural-network system [12], and add the choice of using grade considerations to create custom course recommendations (also see [19]). This second model yields extremely compelling results, but is not capable of the broad range of inference queries possible with our model.

7. CONCLUSION & FUTURE WORK

We have presented a new probabilistic model that is capable of capturing joint relationships between course enrollments, while also allowing for powerful inference queries. There is, however, at least one important drawback to our approach: the strictly Markovian character of the model. Although this assumption allows us to easily learn model parameters, in practice the enrollments observed at one timestep will impact those sampled at the next timestep. Because of this inductive bias our approach is effective with less training data than, for example, a recurrent neural net, and is therefore more easily deployed for institutions smaller in size than our case university.

We emphasize the potential for future work that links data of the sort investigated here with other rich information, such as demographic information describing students, and earned grades. Models incorporating such information could meaningfully identify differences between course trajectories of particular kinds of students, providing insights into how academic policies and programs might be tuned to benefit specific constituencies.

8. REFERENCES

- [1] R. Al-Shabandar, A. Hussain, R. Keight, A. Laws, and T. Baker. The application of gaussian mixture models for the identification of at-risk learners in massive open online courses. *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2018.
- [2] T. R. Bailey, S. S. Jaggars, and D. Jenkins. *Redesigning America's community colleges*. Harvard University Press, 2015.
- [3] R. Baker. Understanding college students' major choices using social network analysis. *Research in Higher Education*, 59(2):198–225, Mar 2018.
- [4] G. Balakrishnan and D. Coetzee. Predicting student retention in massive open online courses using hidden markov models. *Electrical Engineering and Computer Sciences University of California at Berkeley*, 2013.
- [5] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.
- [6] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.
- [7] C. Cremer, X. Li, and D. Duvenaud. Inference suboptimality in variational autoencoders. *arXiv preprint arXiv:1801.03558*, 2018.
- [8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.
- [9] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.
- [10] J. Gardner and C. Brooks. Student success prediction in moocs. *User Modeling and User-Adapted Interaction*, 28:127–203, 2018.
- [11] M. Hlosta, Z. Zdrahal, and J. Zendulka. Ouroboros: early identification of at-risk students without models based on legacy data. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*, pages 6–15. ACM, 2017.
- [12] W. Jiang, Z. A. Pardos, and Q. Wei. Goal-based course recommendation. *CoRR*, abs/1812.10078, 2018.
- [13] A. A. Kardan, H. Sadeghi, S. S. Ghidary, and M. R. F. Sani. Prediction of student course selection in online higher education institutes using neural network. *Computers & Education*, 65:1–11, 2013.
- [14] E. S. Khorasani, Z. Zhenge, and J. Champaign. A markov chain collaborative filtering model for course enrollment recommendations. *2016 IEEE International Conference on Big Data (Big Data)*, pages 3484–3490, 2016.
- [15] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [16] Z. Kovacic. Early prediction of student success: Mining students' enrolment data. 2010.
- [17] B. Motz, T. A. Busey, M. E. Rickert, and D. Landy. Finding topics in enrollment data. In *EDM*, 2018.
- [18] A. Nandeshwar and S. Chaudhari. Enrollment prediction models using data mining. *Retrieved January*, 10:2010, 2009.
- [19] Z. A. Pardos, Z. Fan, and W. Jiang. Connectionist recommendation in the wild. *arXiv preprint arXiv:1803.09535*, 2018.
- [20] Z. A. Pardos and A. J. H. Nam. A map of knowledge. *CoRR*, abs/1811.07974, 2018.
- [21] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [22] D. Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, pages 827–832, 2015.
- [23] J. Scott-Clayton. The shapeless river: Does a lack of structure inhibit students' progress at community colleges? ccrc working paper no. 25. assessment of evidence series. *Community College Research Center, Columbia University*, 2011.
- [24] Q. Song and B. S. Chissom. New models for forecasting enrollments: Fuzzy time series and neural network approaches. 1993.

A Human-Machine Hybrid Peer Grading Framework for SPOCs

Yong Han Wenjun Wu Suozhao Ji
hanyong@nlsde.buaa.edu.cn wwj@nlsde.buaa.edu.cn jisuo Zhao@nlsde.buaa.edu.cn
u.cn

Lijun Zhang Hui Zhang
ljzhang@nlsde.buaa.edu.cn hzhang@nlsde.buaa.edu.cn
u.cn

State Key Laboratory of Software Development Environment,
School of Computer Science, Beihang University, China

ABSTRACT

Peer-grading is commonly adopted by instructors as an effective assessment method for MOOCs (Massive Open Online Courses) and SPOCs (Small Private online course). For solving the problems brought by varied skill levels and attitudes of online students, statistical models have been proposed to improve the fairness and accuracy of peer-grading. However, these models fail to deliver accurate inference in the SPOCs scenario because affinity among students may seriously affect the objectivity and reliability of students in the peer-assessment process. To address this problem, this paper proposes a human-machine hybrid peer-grading framework, including an automatic grader to ensure reasonable peer grades before the Bayesian models are utilized to infer the true scores. This framework can significantly eliminate the severely biased grades by those undutiful students, and thus improve the accuracy of the true-score estimation in the Bayesian peer-grading models. Both simulated and real peer-grading datasets in our experiments demonstrate the effectiveness of this new framework for SPOCs.

Keywords

peer grading, human-machine hybrid algorithm, Bayesian model, auto-grader, SPOCs

1. INTRODUCTION

SPOCs is a version of MOOCs used locally with on-campus students. Despite the difference between SPOCs and MOOCs that SPOCs has the relatively smaller number of students than a MOOCs course [8], a SPOC course needs the same peer-grading process as a MOOC course when the instructor has to evaluate hundreds of open-ended essays and exercises

such as mathematical proofs and engineering design problems within a deadline.

Previous research efforts on peer-grading suggest that there is a great disparity between the observed scores presented by student graders and the true scores given by the instructor. This is because students sometimes can't perform grading tasks as a professional instructor with the right skill and dedication. In the process of peer grading of SPOCs, every student grader needs to submit his answer to the problems of home assignments, and evaluate other peer's submissions according to the rubrics provided by the course instructor. The previous models [7][6] mainly adopt a Bayesian-based approach by considering the major factors affecting the aggregation of peer graded scores including the bias and reliability of every student grader.

These peer grading algorithms mostly designed for MOOCs courses may have poor performance in the setting of SPOC courses because they ignore another important factor – student attitude toward their grading tasks. Due to affinity among students in a SPOC course, they tend to assign random scores to other submissions without seriously evaluating their peers' homework. Even worse, in our real experiment, we found that some students simply give a full score to every submission assigned to them. Therefore, such an undutiful grading behavior violates the basic assumption in those Bayesian statistical models and unavoidably generate data noises that severely degrade the performance of the models. Our simulation and real experiment confirm that the models produce inaccurate estimations for final scores in the process of per grading [3].

To address the problem, this paper proposes a novel human-machine hybrid framework that combines assessment effort of both human and machine for peer-grading. The framework adopts a document classifier as an auto-grader that evaluates students' submissions to estimates their scores, and compares the scores with the peer-graded scores. Then, it attempts to filter out the unreasonable peer-graded scores that are significantly different from its estimations, and retain these legitimate scores for the statistical models. In this way, it can alleviate the negative impact of student ran-

Yong Han, Wenhyn Wy, Suozhao Ji, Lijun Zhang and Hui Zhang
"A Human-Machine Hybrid Peer Grading Framework for SPOCs" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 300 - 305

dom grading behavior and improve the overall performance of peer-grading models. Experimental results on the actual and simulation datasets demonstrate that our hybrid framework outperform the original peer-grading models in terms of the true-score estimation accuracy without placing too much extra workload on course teaching assistants (TAs).

The rest of paper is organized as follows: Section 2 discusses the related work of our research. Section 3 elaborates the main problems of current models in the peer grading of our SPOCs and explains the motivation of combining the machine and the human effort in peer grading. Section 4 describes the design of the human-machine hybrid framework for peer-grading in detail. Section 5 presents our experimental results.

2. RELATED WORK

The focus of this paper is to combine the power of human graders and a machine grader to improve the predictive ability of the existing peer grading models. Numerous papers have been published on the field of peer-grading research. Most researchers attempt to tackle with the peer-grading problems from the two aspects: statistical methods for accurately inferring true scores and incentive mechanism to motivate and regulate student grading behaviors.

One of the major research topics in peer-grading is to build a Bayesian statistical model that can accurately infer the true-scores of student submissions. Such models were proposed in [7] and [6] for peer-grading in MOOC courses with bias and reliability of student graders as the major latent factors. In [10], Ueno utilize Item Response Theory to model the score estimation, difficulty of problem and a grader's capability as parameters in the IRT equation. The major limitation of these models is caused by their assumption that every student follows a statistical model in the peer-grading process. But in practice, especially in the scenario of SPOCs, students' grading behavior actually are heavily affected by their motivation and attitude towards peer-grading tasks. Some students grade homework in a dutiful manner whilst others simply assign scores randomly. Thus, a single statistical model cannot describe all the possible grading strategies among these students in a SPOC course.

The problem of student grading behavior has received attention from academic researchers in the field of game theory. Recently, peer prediction mechanism has been proposed to incentivize truthful reports from individual students in the process of peer-grading [3][1]. Without the ground truth scores for every submission to verify against, designers of peer prediction mechanism often introduce comparison algorithms that compare grading results among multiple student graders and enforce penalties on those whose evaluation outcomes are different from their peers. But peer-prediction has its inherent limitation because there are potentially multiple Nash equilibria where students might be able to coordinate to avoid penalty without revealing their informative signal truthfully. Even when the peer-prediction mechanisms do offer a truthfully equilibrium, they also always induce other uninformative equilibria [2]. In the settings of SPOCs, affinity among students make it highly possible for them to collude in the peer-grading process to cheat the peer-prediction mechanisms.

Our human-machine hybrid framework is complementary to the research efforts on the statistical peer-grading models and spot-checking mechanisms of peer-prediction. The auto-grader in our framework can help to eliminate unreliable assignment grades so as to ensure only quality grades are passed onto the statistical models such as the PG family model. In this way, the auto-grader can be adopted in spot-checking mechanisms and work as an online supervisor to perform checking tasks on behalf of TAs and update TAs with its screening results.

The development of reliable auto-grader is widely regarded as a challenging task. Many researchers such as [9][5] designed neural network-based auto-graders to evaluate open essays. The state-of-art automatic graders can't complete grading tasks in a full autonomous way, especially for science essays and technical reports in domain-oriented courses. Thus, our framework only assumes an automatic grader with limited classification capability and regard it as an intelligent assistant that can work with course instructors and TAs in the process of peer-grading.

3. PROBLEM ANALYSIS OF PEER GRADING MODELS

In the section, we first introduce the peer grading (PG) models, then discuss the problems of the PG models when they are applied in the SPOC settings. Through the simulation experiment, we analyze fault tolerance of the PG models with the increase in the number of undutiful students.

3.1 Peer Grading Models

We apply the PG models [7][6][4] in the SPOCs scenario, which are Bayesian graph models with the latent factors including the biases and reliabilities of the peer graders. These models of Eq (1)(2)(3)(4) define z_u^v the observations grade which is affected by the latent factors including b_v , τ_v , and the learner's true grade s_u . The parameter ρ denotes factors that affect the reliability, and the remaining parameters β , η , μ , γ , λ in Eq (1)(2)(3)(4) are hyper-parameters.

$$\tau_v \sim \mathcal{N}(\rho, 1/\beta_0) \quad (1)$$

$$b_v \sim \mathcal{N}(0, 1/\eta_0) \quad (2)$$

$$s_u \sim \mathcal{N}(\mu_0, 1/\gamma_0) \quad (3)$$

$$z_u^v \sim \mathcal{N}(s_u + b_v, \lambda/\tau_v) \quad (4)$$

3.2 Limitations of the PG models in SPOCs

There are two major factors that may prohibit SPOCs students from performing peer-grading tasks in a fair and accurate way. First, students without the right knowledge and dedication may regard peer-grading tasks as unnecessary burdens and decide to give the assignments random scores. Second, affinity among SPOCs students who often interact with each other in the same campus or even classroom may drive them to assign higher grades to her or his peers' submissions. Both factors can result in high deviation between the observation grades z_u^v and the ground-truth grade. We run the simulation experiment to evaluate the impact of student's attitude of peer assessment and analyze the tolerance of the PG models against data error generated by student graders. Based on the configuration of the simulation, we extend the PG models as follows: Assume that each student becomes a dutiful or an undutiful students with a certain

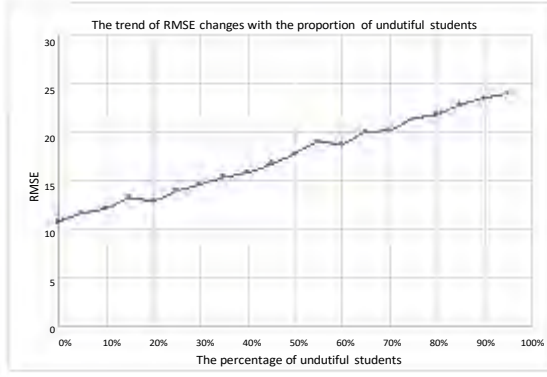


Figure 1: The correlation between the RMSE of the PG model and the proportion of undutiful students in the simulation.

probability each time they review. Define the number of students with undutiful grading attitude as $p \in [1, n]$. Although the value of p can also be 0, here we define the value of p starting from 1 for the convenience of calculation, and the n is the total number of all students. Define a grading strategy set D where every element $d_i \in D$ denotes a particular distribution corresponding to the strategy to follow. The set D contains the distributions (5) and (6):

$$z_u^v \sim \mathcal{N}(s_u + b_v, \lambda/\tau_v) \quad (5)$$

$$z_u^v = x \pm \text{random}(y) \quad (6)$$

The Eq (5) represents the strategy distribution in which the observed scores are presented by the good students with dutiful grading attitude, and the Eq (6) represents the other strategy distribution in which the observed scores are presented by the undutiful students with high deviation. In Eq (6), x is the set to the average grading scores based on experiences and y is set to an random value with the range $[0, 20]$. For simplicity, we assume that a student determine his/her choice of the grading strategy before he accepts the grading task and will not change in the middle of the grading process.

Figure 1 shows that the RMSE of the prediction grades has a linear correlation with the proportion of undutiful peer graders and its value ranges in $[10, 25]$. This result remains even when we change the parameters (x, y) in the Eq (6). The expression of RMSE can be defined in Eq (7), where $X_{model,k}$ denotes the specific prediction grade prediction generated by the PG models for an exercise report k , and $X_{true,k}$ denotes the corresponding ground truth score of the exercise report k .

$$RMSE = \sqrt{\frac{\sum_{k=1}^n (X_{model,k} - X_{true,k})^2}{n}} \quad (7)$$

We can expand the Eq (7) by separating the errors generated by the dutiful group and undutiful group. First we define $e_k = X_{model,k} - X_{true,k}$ ($k \in [1, n]$), then we define $p\bar{e} = \sum_{i=1}^p e_i$ ($p \in [1, n]$) denotes the sum of the set $A = \{e_i | i \in [1, p]\}$ and $(n-p)\bar{f} = \sum_{j=p+1}^n e_j$ denotes the sum of the set $B = \{e_j | j \in [p+1, n]\}$. So, we transform the Eq (7) into Eq (8) on the condition that each element in A and B are

equal,

$$RMSE \cong \sqrt{\frac{p(\bar{e}^2 - \bar{f}^2)}{n}} + \bar{f}^2 \quad (8)$$

Because of the assumption $|\bar{e}| \geq |\bar{f}|$, the value of RMSE increases with p changing from 1 to n . Thus we can summarize that the grading attitude of the students can significantly affect the performance of the PG model.

3.3 Comparison among grading error distributions in the simulation and actual datasets

By comparing different inference performance of the PG models in both simulation experiments and the real dataset, we analyze the effect of the features of bias b_v and reliability τ_v on the precision of inferring true score s_u . In the Gibbs sampling process for fitting the PG models, the Eq (9) updates s_u in iterations. where the variable z_u^v is a constant value, besides the τ_v and b_v , the others are hyperparameters. From Eq (9) we can infer that the main factors affecting the true grade s_u include a grader's bias and reliability.

$$s_u \sim \mathcal{N}\left(\frac{\gamma_0 \mu_0 + \beta_0 \tau_{u_i} + \sum_{v:v \rightarrow u_i} \frac{\tau_v (z_u^v - b_v)}{\lambda}}{\gamma_0 + \beta_0 + \sum_{v:v \rightarrow u_i} \frac{\tau_v}{\lambda}}, \frac{1}{\gamma_0 + \beta_0 + \sum_{v:v \rightarrow u_i} \frac{\tau_v}{\lambda}}\right) \quad (9)$$

In order to verify the conclusion of our analysis, we compare the grading errors of the PG models in simulation experiments and the real dataset. The real dataset was collected in the SPOC course on Computer Network in our university. We build an online learning system to support the session of the course with the total enrollment of 724 students. Figure

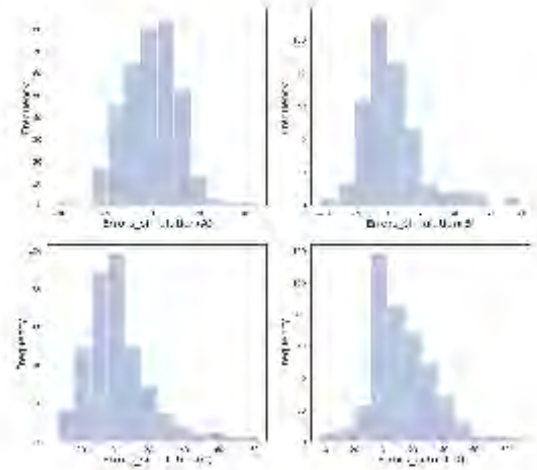


Figure 2: The distributions of errors in three simulation datasets and the actual dataset. Fig A, B and C denote the histogram of grading errors generated by simulation experiments. Fig D denotes the distribution of the real dataset based on the Computer Network course.

2 shows that the simulation and actual datasets have a very different error distribution. Fig 2A assumes that every student's grading behavior follows the gaussian model defined

in Eq (5). In contrast, the real dataset in Fig 2D indicates that many students' grading behavior doesn't satisfy the gaussian distribution. In order to further confirm the conclusion, we have conducted the other simulation experiments, in which we configure 40% undutiful students and 60% undutiful students to follow the random grading behavior defined in Eq (6), respectively. The results as shown in Fig 2B and Fig 2C demonstrates a similar error range to Figure 2D. These observations suggest that students in the SPOC experiment tend to exhibit random grading behavior. Clearly, such a high deviation of the peer grades in the real dataset from their ground truth is the reason why the PG models cannot achieve the low RMSE as we expect.

4. THE HUMAN-MACHINE HYBRID PEER GRADING FRAMEWORK

This section presents the design of our human-machine hybrid framework in detail, as shown Figure 3. The main idea of the framework is to use the auto-grader as an anomaly detector to screen the peer grades generated by undutiful students. The framework consists of three major components including a homework Auto-Grader, a Score-Filter and the PG models.

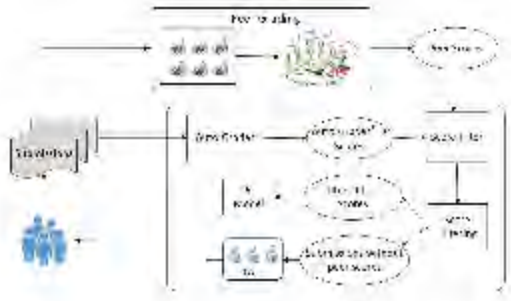


Figure 3: The human and machine hybrid framework of peer grading.

In the process of peer grading, the system first allocates the tasks for each student to perform their peer-grading tasks. After the Auto-Grader receives a score for a submission, it estimates a score for the same submission, and passes the estimation to the ScoreFilter. The ScoreFilter is responsible for comparing the Auto-Grader's estimation with the original peer score, and abandoning the peer score if the deviation between these two scores goes beyond the predefined threshold. With the co-ordination of the Auto-Grader and the ScoreFilter, the framework divides the student submissions into two groups: one group includes the submissions with legitimate peer grades that can be aggregated by the PG model for the grade inference, the other includes those without valid peer grades that have been sent to TAs for evaluation.

4.1 Naive Bayesian based Classifier as Auto-Grader Implementation

Based on Naive Bayesian method, we design a weak text classifier as the Auto-Grader in the hybrid peer grading framework. Each course assignment report often contains several problems. Thus the Auto-Grader's design consists

of several classifiers, each of which classifies one problem in the assignment report. The grade classification results for all the problems of the assignment are mapped into scores based on its rubric and combined together as the total score of the assignment report.

4.2 Score Filtering and Postprocessing

The ScoreFilter in the hybrid human-machine grading framework adopts a simple filtering process. It computes the absolute value of the difference between grades estimated by the Auto-Grader and the peer-graded scores, sorts the scores in a descending order, and filter out the top 20% with the highest deviation values. The design of the score filter involves two major issues: The threshold for dropping unreasonable scores and the post-processing strategy for supplementing abandoned scores.

The Error Threshold of Score Filtering

Because our Auto-Grader is a weak classifier, we need to consider the classification error of each sub-problem of a homework report when we use the Auto-Grader to evaluate each sub-problem. We define the following equation to calculate the grading error.

$$Threshold_{error} = \sqrt{\frac{\sum_{i=1}^n (x_i - a_i)^2}{n}} \quad (10)$$

In the Eq (10), $x_i \in x_1, x_2, \dots, x_n$ denotes the score given by a student grader, $a_j \in a_1, a_2, \dots, a_n$ denotes the score estimated by the Auto-Grader. The value of n presents the number of the problems in an assignment. We use Eq (10) to predict the error for each peer-graded score, and sort the list in a descending order according to the value of the prediction error, thus filtering out the peer grades with high errors values.

The Post-Process Strategy of Score Filtering

This simple filter algorithm above may cause potential problems for the PG models. After the ScoreFilter drops these unreasonable peer-graded scores, it can create extreme cases where most peer scores for a student assignment are eliminated. In such a case, a post-processing step is necessary in the ScoreFilter to supplement new scores for the downstream PG models. For the post-processing step, we propose



Figure 4: Three Strategy to replace filtered grades.

three strategies to handle the filtered-out scores. Dropping only: The ScoreFilter simply drops the scores identified by the auto-grader and does not supplement any new scores; Replacement by Auto-Grading: The Score-Filter directly uses the grades generated by the Auto-Grader to replace the peer scores that are identified as biased; Mixed Replacement: This strategy is only designed as a contrast strategy, which can choose the replacement score for a filtered peer score among the rest peer scores and the score predicted by the auto-grader based on their absolute difference value from the ground truth. Although it is impossible to implement

this strategy in the real system, it gives us an upper-bound for the strategy design when the ground-truth is available.

Figure 4 presents the example of all the strategies. In Figure 4, the leftmost graph is the relationship between the original peer score and the real score of the submission, from left to right, the second subgraph represents the score aggregation method using the first strategy, and the third subgraph Represents the score aggregation method using the second strategy, and the last subgraph represents the using of the third strategy for score aggregation.

5. EXPERIMENTS AND RESULTS

The peer-grading experiment was conducted in the course of Computer Network, which is offered to the senior college students of the computer science major. After class, students must design a networking plan and describe device configurations in their laboratory reports. These reports are evaluated through the peer-grading process. Our experimental dataset was collected from the class sessions in Year 2015-2017, including a total of 6 peer grading assignments and 724 students and 2354 assignment reports.

5.1 The prediction accuracy of the Auto-Grader

We choose the assignment reports on the sub-networking chapter of the course as the training and test data to develop the classifier of the Auto-Grader. This sub-networking assignment consists of six problems. For each problem in the assignment, there is a rubric specifying the grading category and score scheme. Table 1 displays the categories of rubric for each problem.

Table 1: The categories of the each problem of the assignment.

Problem ID	Category 1	Category 2	Category 3	Category 4
1, 2, 3	0	5	10	—
4	0	10	20	—
5	0	10	15	20
6	0	10	—	—

In the rubrics for Problem 1-3, there are three categories and the scoring values of each category are 0, 5 and 10 points. The rubric for Problem 4 also has three categories, including 0, 10 and 20 points. The rubric for Problem 5 has 4 categories, including 0, 10, 15, and 20 points. The rubric for Problem 6 only has two categories, including 0 and 10 points. Based on the above design of rubrics, the classifier of our Auto-Grader can achieve reasonable grading accuracy. The experimental results of the Auto-Grader are shown in Table 2. The grading accuracy of the Auto-Grader classifier

Table 2: The prediction accuracy of Auto-Grader based on Naive Bayes.

Problem ID	≤ 5	≤ 10
1	66.29%	100%
2	73.6%	73.6%
3	73.03%	73.03%
4	60.11%	90.45%
5	66.85%	87.64%
6	65.73%	100%

within 5 points can achieve more than 60%, and the accuracy within 10 points becomes higher partly because of the design of the rubrics. This shows that the Auto-Grader can present reasonable score estimation as long as the threshold of the error is set to 10 points.

5.2 Choice of Post-processing Strategies for Score Filtering

We evaluate the performance of the ScoreFilter, especially the post-processing strategy. In addition to the three strategies described in Section 4.2, we also run the post-processing with the ground-truth strategy, in which the filtered top 20% peer scores are replaced by the ground-truth value. From Table 3, one can find that the Dropping-only strategy shows better performance than the Replacement by Auto-Grading strategy. The reason may be caused by the limited grading accuracy of the classifier in the Auto-Grader. Although the Mixed-Replacement strategy and the Ground-truth strategy achieve the lowest RMSE, their implementation is not feasible in the real scenario. Therefore, we have chosen the Dropping-Only strategy for post-processing in the ScoreFilter.

Table 3: The value of RMSE of Adopting the three post-processing strategies.

Post-Processing Strategy	RMSE	Post-Processing Strategy	RMSE
Dropping only	17.29	Mixed Replacement	16.45
Replacement by Auto-Grading	30.89	Only Ground Truth	15.96

5.3 Tuning the Threshold of the Score Filter

When the Naive Bayesian based auto-grader is used to each problem in an assignment submission, we need to consider the classification error of each sub-problem when we use auto-grader to evaluate the grade of each sub-problem by Eq (10).

Tuning the error thresholds

We investigated the impact of the error threshold by comparing the value of RMSE generated by the PG models and the Auto-Grader under different threshold values. The results are shown in Table 5. In Figure 5, we set the threshold

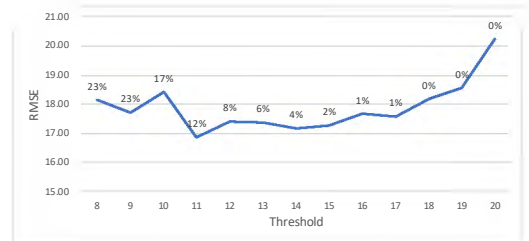


Figure 5: The trend of RMSE changes with threshold. The online labels indicate the percentage of submissions that do not have a peer score as a percentage of the total of submissions.

to filter the number of peer-graded scores with an interval of 1. We found that when the threshold is 11, the value of

RMSE drops to the lowest value, but the number of the submissions without peer grades accounts for 12% of the total submissions. In this case, the class TAs have to check these submissions and give their evaluations as the input for PG model. Therefore, when the number of assignment report is large, it will bring extra workload to the class TAs. Through our further experiments, we find optimal threshold should be 14.3, where the minimum RMSE can be calculated as 16.38, and only 3% submissions have to be assessed by the class TAs. In practice, the class instructors have to run a few rounds of peer-grading to determine the distribution of peer-grades scores and set the empirical value for the error threshold.

5.4 Overall Performance of the Hybrid Peer-Grading Framework

In order to evaluate the performance of the hybrid peer-grading framework, we run the PG models after the peer-graded scores are filtered by either the framework or random filtering respectively. In this way, we can generate three group of experimental data: the initial dataset without any score filtering, the dataset with Naive Bayesian-based Auto-Grader filtering, and the dataset with random filtering. The RMSE of the PG models based on the three data sets is shown in table 4.

After the peer-graded scores are sorted in a descending order of the estimated error, the top 20% of the scores are filtered out in each experiment. The filter process may eliminate all peer-graded scores for some submissions which have be re-evaluated by TAs. In the above experiment, when the error threshold is set to 15, 706 submissions are left with at least one peer-graded score. Only 18 submissions which account for 2% of all lose all the peer-graded scores. Thus, the task of re-evaluating these submissions does not bring too much burden to the course TAs. It can be seen from the Table 4, no matter which PG model is used, the human-machine hybrid framework can obtain the best performance, which averagely reduces the RMSE by 4. This outcome confirms that the hybrid human-machine peer-grading framework can improve prediction accuracy of the PG models with the presence of random grading behavior.

6. CONCLUSION

In this paper, we introduce a novel human-machine hybrid peer-grading framework to alleviate the problem of the random grading where student graders perform their peer-grading tasks in an undutiful manner. The most important component of the framework is the Auto-Grader that can classify

students' submissions using machine learning and enable the framework to filter out the peer-graded scores with high errors. When filtering the peer grades, the framework calculates the error threshold according to the RMSE metric. Extensive experiments confirm that the hybrid framework can effectively eliminate the noise in peer-graded scores made by undutiful student graders and improve the prediction accuracy of the PG models.

7. ACKNOWLEDGMENTS

This work was supported by grant from the National Key Research and Development Program of China (Funding No. 2018YFB1004502) and supported by NSFC (Grant No. 61532004), and Grant from State Key Laboratory of Software Development Environment (Funding No. SKLSDE-2017ZX-04).

8. REFERENCES

- [1] L. De Alfaro, M. Shavlovsky, and V. Polychronopoulos. Incentives for truthful peer grading. *arXiv preprint arXiv:1604.03178*, 2016.
- [2] A. Gao, J. R. Wright, and K. Leyton-Brown. Incentivizing evaluation via limited access to ground truth: Peer-prediction makes things worse. *arXiv preprint arXiv:1606.07042*, 2016.
- [3] X. A. Gao, A. Mao, Y. Chen, and R. P. Adams. Trick or treat: putting peer prediction to the test. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 507–524. ACM, 2014.
- [4] Y. Han, W. Wu, and X. Zhou. Improving models of peer grading in spoc. In *EDM*, 2017.
- [5] N. Madnani and A. Cahill. Automated scoring: Beyond natural language processing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1099–1109, 2018.
- [6] F. Mi and D.-Y. Yeung. Probabilistic graphical models for boosting cardinal and ordinal peer grading in moocs. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [7] C. Piech, J. Huang, Z. Chen, C. Do, A. Ng, and D. Koller. Tuned models of peer assessment in moocs. *arXiv preprint arXiv:1307.2579*, 2013.
- [8] Y. Song, Z. Hu, and E. F. Gehringer. Collusion in educational peer assessment: How much do we need to worry about it? In *2017 IEEE Frontiers in Education Conference (FIE)*, pages 1–8. IEEE, 2017.
- [9] K. Taghipour and H. T. Ng. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891, 2016.
- [10] M. Uto and M. Ueno. Item response theory for peer assessment. *IEEE transactions on learning technologies*, 9(2):157–170, 2016.

Table 4: RMSE comparison between the human-machine framework and the PG models.

Models	RMSE		
	Without Auto grader Filtering	With Naive Bayes-based Auto-grader Filtering	Generated by filtering randomly
PG1	21.90	17.09	22.10
PG3	20.40	17.30	21.36
PG4	21.57	17.49	22.02
PG5	20.26	16.71	21.86

Modelling End-of-Session Actions in Educational Systems

Christian Hansen, Casper Hansen, Stephen Alstrup, Christina Lioma
University of Copenhagen
Department of Computer Science, Denmark
{chrh,c.hansen,s.alstrup,c.lioma}@di.ku.dk

ABSTRACT

In this paper we consider the problem of modelling when students end their session in an online mathematics educational system. Being able to model this accurately will help us optimize the way content is presented and consumed. This is done by modelling the probability of an action being the last in a session, which we denote as the End-of-Session probability. We use log data from a system where students can learn mathematics through various kinds of learning materials, as well as multiple types of exercises, such that a student session can consist of many different activities. We model the End-of-Session probability by a deep recurrent neural network in order to utilize the long term temporal aspect, which we experimentally show is central for this task. Using a large scale dataset of more than 70 million student actions, we obtain an AUC of 0.81 on an unseen collection of students. Through a detailed error analysis, we observe that our model is robust across different session structures and across varying session lengths.

Keywords

Educational session modelling, Student behaviour, Deep learning

1. INTRODUCTION AND RELATED WORK

Digitization of education is ever increasing, where for higher level education, massively open online course (MOOC) platforms are offering an increasing amount of high quality courses for a wide range of topics. Similarly, a wide variety of systems exist for assisting teachers in lower levels of education, where especially mathematics has been a focus since many types of exercises allow automatic correction, thus freeing up teacher resources. The design of most of these systems is that students engage with them through sessions consisting of different actions, e.g. answering questions or reading learning material, but the frequency and length of sessions naturally vary between students.

In this paper we consider the problem of modelling when an action is the last action in a session, which we denote as the End-of-Session action. This is naturally a highly class imbalanced problem, since there is only one End-of-Session action per session, and we do not get any explicit feedback about the student's intent to end a session before it is over. Additionally, a large amount of external and internal factors for ending a session exists, for example: The student has finished an assigned task (e.g. homework), the student is unmotivated or distracted, the student has grasped the material, etc. If we are able to model the End-of-Session probability for each action accurately, then it can offer real-time insights in ongoing student sessions, and actions can be taken to steer the student in a direction where she is less likely to quit.

The problem of modelling when a session ends has, to the best of our knowledge, only recently been considered by Kasak et al. [8] in the educational domain, where a polynomial classification model with handcrafted features was used, but the work was done on a small scale using just 452,000 student actions. Related work to this problem has also been considered from the point of view of modelling and clustering student sessions, in order to understand and group student behaviour in educational systems [11, 5, 9]. Even though the End-of-Session problem has not been investigated much in the educational domain, the problem has been considered in other domains, such as business [13] and media streaming [14, 3], where recurrent neural networks and gradient boosted trees have been shown to work well.

A related problem to End-of-Session modelling that has been considered extensively in the educational domain, is drop-out prediction. In this setting student log data is used for modelling drop-out of students, with studies focusing on whether a student will drop out of their studies [1], or drop out of MOOC courses [2, 10, 15]. Drop-out prediction models can utilize both general characteristics of a student, as well as usage behaviour changes (e.g. usage decline) to aid in the prediction. Although End-of-Session modelling utilizes much of the same types of data, in this setting the "stop"-signal needs to be found within each session, and not as a single terminal event happening at a single point.

In this paper we model the End-of-Session problem using a deep recurrent neural network architecture, that outputs an End-of-Session probability for each student action, using the actions in the current and previously completed sessions.

Christian Hansen, Casper Hansen, Stephen Alstrup and Christina Lioma "Modelling End-of-Session Actions in Educational Systems" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 306 - 311

We show that considering the task on a studentwise level obtains significant improvements over considering the sessions individually. We experimentally evaluate our approach using log data from more than 70 million student actions from 85,780 students, where we are able to obtain an AUC score of 0.81 on an unseen collection of students. Through a detailed error analysis we show that the model is robust across students with a varying number of completed sessions, across sessions of different lengths, as well as for different session structures.

In the remaining of the paper the data is presented in Section 2; the End-of-Session problem is presented in Section 3, and our deep learning approach presented in Section 3.1; experimental findings are presented and analyzed in Section 4; and a conclusion is made in Section 5.

2. DATA

In this paper we use data provided by Edulab¹, the largest Danish system for online mathematics. The system primarily targets students aged 6 to 16 and supplies material spanning the entire mathematics curriculum. The system offers two primary ways of engaging with the material: 1) Reading text material or watching video material, and 2) answer fill-out or multiple-choice questions. These two broad categories can be done by students on their own initiative or by being assigned it as homework by a teacher. All questions and video/text materials are associated with a specific *lesson*, which is a specific skill such as "addition of small integers", and each lesson is associated with a *topic*, which is a broad skill such as "addition". The general data statistics can be seen in Table 2.

We use log data generated when students use the system. For each action they perform we have the timestamp, the lesson id, the topic id, the type of action (and an answer if the action is answering a question), and lastly whether it is homework. The system does not track the exact time taken to complete an action, but can derive it as the time taken since the last action. To group the actions into sessions, we consider a time gap of more than 15 minutes to be a new session. The choice of 15 minutes was based on each action rarely requiring more than a few minutes, and to allow time for breaks within the same session. From these we derive a set of general features associated with each action as described in Table 1. For the temporal features we chose to discretize the time into the time intervals 8-12, 12-15, and 15-8 in order to represent the times as "before noon", "afternoon", and "after school". For content type we decided to ignore the actual ids, and instead focus on the changing between lessons and topics. The reason for this was the large number of different ids, with close to 1000 different lesson ids. This also has the benefit of being more general, since the system will not be able to learn patterns specific to certain ids.

There is a large variance in the number of sessions each student has completed and the length of each of the sessions, the histograms for each of these can be seen in Figure 1 and 2 respectively. For the histogram of session lengths, spikes are seen at certain lengths divisible by 5, this is due to how

¹The data is proprietary and not publicly available.

Category	Features
Temporal	1) time of day as discrete values 8-12, 12-15, and 15-8, 2) time since last action, and 3) time since last session.
Action type	1) answering fill-out question, 2) answering multiple-choice question, and 3) watching or reading material.
Content type	1) the lesson id was changed compared to the previous action, and 2) the topic id was changed compared to the previous action.
Miscellaneous	1) whether a potential question is answered correctly, and 2) if the action was done as homework or on the student's own initiative.

Table 1: Dataset features divided into categories.

Number of students	85,780
Number of sessions	2,587,876
Number of actions	71,341,770
Percentage of sessions being homework	48.3%
Percentage of sessions being partly homework	25.5%
Percentage of sessions not being homework	26.2%

Table 2: General data statistics.

some groups of questions are presented to the students, e.g. as 5 or 10 quick related questions. Similarly, homework often consists of a number of questions divisible by 5.

3. MODELLING END-OF-SESSION

In this section we present the problem of modelling when a session ends, by predicting the probability of a certain action being the last in the current session, which we denote as the End-of-Session action. A session consists of a number of actions being either reading or watching some material M or answering a question Q , for example:

$$M \rightarrow M_{\text{lesson changed}} \rightarrow M_{\text{lesson changed}}^{\text{topic changed}} \rightarrow \quad (1)$$

$$Q \rightarrow Q \rightarrow Q_{\text{lesson changed}} \rightarrow M$$

where the sub- and superscript denote if the lesson or topic id was changed compared to the previous action. To each of these actions we associate a binary label indicating whether or not the action is the last action in the session. We consider this binary label a probability of an End-of-Session action, such that the last action has a probability of 100% for ending the session and all the previous ones have 0%. Naturally, this leads to a very imbalanced dataset. The goal of this task is to create a model able to assign a probability score to each action indicating its End-of-Session probability, and such that it is largest at the true End-of-Session action.

Usage of the Edulab system is very non-restrictive. A student can engage in any material she wants, and can choose to do the assigned homework at any time. Due to this it is inherently individual how each student uses the system, and when they end their session. It is therefore relevant to consider the problem on a student level, as students have their own preferences with regards to session length, and consequently when the End-of-Session action occurs. This kind of individual behaviour does not only define the general length of student session, but can also influence the dynam-

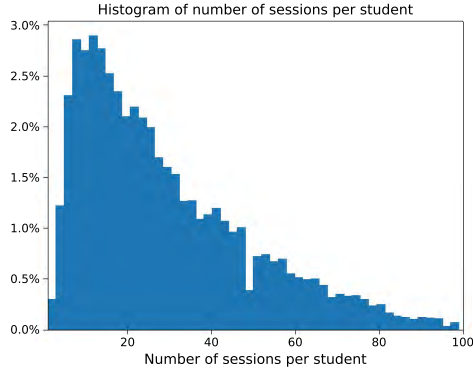


Figure 1: The distribution of sessions per student.

ics between sessions, since e.g. some students may prefer to alternate between longer and shorter sessions; or take a quick repetition session the next time they engage with the system.

3.1 Deep Learning Approach

We propose to use a deep learning approach utilizing a recurrent neural network (RNN) for modelling the End-of-Session problem. This approach has the benefit of not requiring complex feature engineering, and is able to capture complex temporal representations of student behaviour. Certain behaviour patterns may occur at a much earlier point than the current action, so to this end we choose to use a Long Short Term Memory (LSTM) network [7].

Our proposed network architecture is displayed in Figure 3, and can be considered in 4 parts:

- **Actions:** The input to the network are the actions associated with each student. During training each action is labeled either 0 or 1, depending on whether the action is the last in a session. The actions are passed as one long sequence. Each session within this sequence is identifiable by the network through the "time since last session" feature, as well as if the previous label was equal 1, in which case the next action must be a new session.
- **Memory:** The purpose of the memory layer is to let the current interaction's effect on the End-of-Session probability be influenced by previous actions and learned student characteristics. As mentioned previously, we use a LSTM unit for this task, which is able to handle long dependencies in the sequential data, and suffer fewer problems related to vanishing gradients.
- **Fine tuning:** The output of the memory layer is passed to the fine tuning block consisting of two fully connected layers with ReLU activation. The purpose of this is to refine the output of the LSTM, such that the LSTM is able to focus on learning the general underlying behaviour of a student.
- **Prediction:** The fine tuned output is passed to a single neuron with a sigmoid activation. The sigmoid function returns an output between 0 and 1, representing the End-of-Session probability.

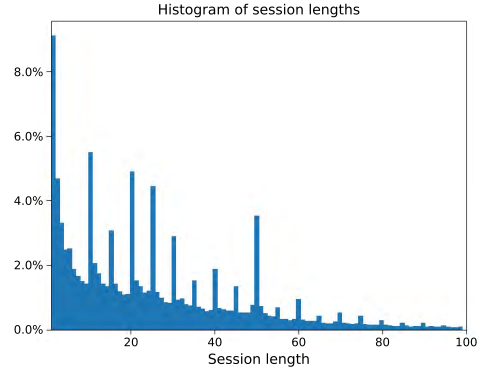


Figure 2: The distribution of session lengths.

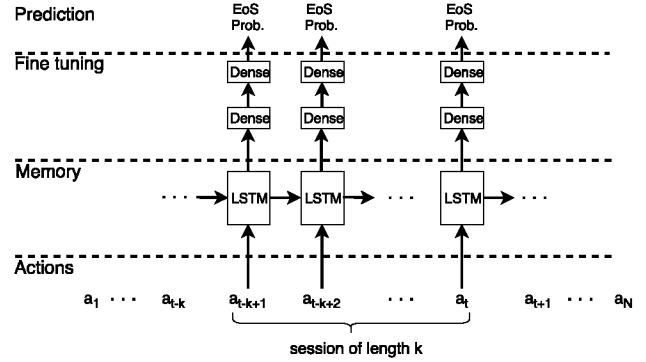


Figure 3: Network architecture of our model.

Between all layers we use Dropout [12], which sets the activation of a neuron to zero with a certain probability, and is used to limit the network's ability to overfit.

3.2 Parameters and training

For the number of neurons in each layer we use a fan-in approach, where the size is halved from each adjacent layer to the next. Due to the training time of the network, extensive tuning of the dropout probability p and layer sizes is beyond the scope of this paper, but initial experiments showed that $p = 0.4$ performed well, which is within the typical range of 0.2 to 0.5 [12]. For the layer sizes, initial experiments showed that a LSTM size of 400 performed well, leading to the fine tuning layers having size 200 and 100 respectively.

We train the network using the RMSprop optimizer with a learning rate of 0.001, and use binary cross entropy as the loss function. Due to the End-of-Session problem being heavily class imbalanced, we do a studentwise re-weighting such that if a student has 400 actions consisting of 16 sessions, then each End-of-Session action is weighted by 25, while the others have a weight of 1. This forces the network to balance its ability to predict both normal and End-of-Session actions. This re-weighting changes from student to student based on each student's average session length.

3.3 Session level model

In the previous section we presented our approach for modelling the End-of-Session probability by utilizing the full history of past actions for each student. As a baseline approach

we can use the same model architecture, but do the modelling on a session level, using just the past actions in the current session, instead of on a student level as before. This will show the benefit of personalizing the model, instead of just focusing on each individual session alone.

4. EXPERIMENTAL EVALUATION

In this section we will describe the experimental evaluation and error analysis of our approach. Our experimental performance comparison is between the performance of the student level model versus the session level model, in order to quantify to what degree a session is self-explanatory with regards to its length, and whether learning a student representation is beneficial. Based on this we analyze the student level model in the following ways:

1. We investigate how the End-of-Session probabilities associated with each action increases or decreases throughout a session.
2. We investigate how homework sessions influence the model's performance, since these can be considered fixed compared to when students use the system on their own initiative.
3. We consider how the model performs for sessions of varied length, in order to investigate the necessary session length for the model to perform well.
4. We consider how the model performs on students with limited system usage, i.e. when considering students with a varying number of sessions, in order to investigate how many sessions are needed for this task.

4.1 Experimental Setup

We split the students in our dataset randomly into a training set, validation set, and testing set. We use 90% of the students for training and the remaining 10% for testing. For validation 10% of the students from the training data are used. Thus, the students validated and tested on have not been seen previously during training. For training the network we use a batch size of 64, and employ early stopping with a patience of 3, i.e. we use the model with the best validation performance, and stop after no improvements have been seen in 3 epochs. We use the same approach for training the session level model, where the sessions are extracted from the students in each of the sets.

For measuring the performance we use the area under the receiver operator characteristic curve (AUC). The reason for this is that we do not require the predicted End-of-Session probabilities to be either 0 or 1 (as their labels), but rather expect them to increase slightly over time, with a large increase in the close proximity of the End-of-Session action. Since the sessions are of vastly varied lengths (see Figure 2), we argue that this measure is very useful, since it handles the class imbalance, and most importantly provides a measure for how we are able to rank normal and End-of-Session probabilities compared to each other across all students.

4.2 Model Performance

In section 3.1 we presented two models: The first being the student level model where we used all previous actions when

Model	AUC
Student level model	0.8103
Session level model	0.5647

Table 3: AUC scores of the student and session level model.

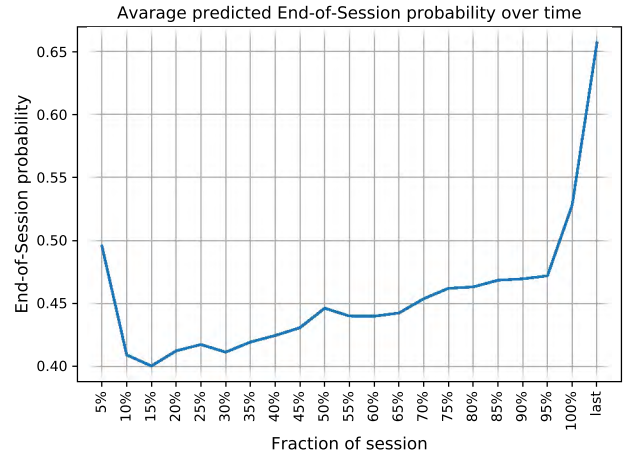


Figure 4: Average End-of-Session probability for sessions split into 5% chunks, done for sessions with at least 20 actions.

predicting the End-of-Session probability of an action, and the second being the session level model where we considered each session alone. We do not expect the session level model to perform as well as the student level model, but this comparison will show to what extent a student's current behaviour can be described by just the current session. Table 3 displays the AUC scores for each of the models. The student level model significantly outperforms the session level model (AUC of 0.8103 vs 0.5647 respectively), showing the importance of modelling the student.

We will now consider how the End-of-Session probabilities associated with each action increase or decrease throughout a session. To do this we consider all sessions with at least 20 actions, and cut each session into intervals consisting of 5% of the actions. This means that each interval for a session with length 20 consists of 1 action, while one of length 100 has 5 actions in each interval. For each of these intervals we compute the average End-of-Session probability. We do this for all sessions and plot the average End-of-Session probabilities for each of the intervals in Figure 4.

Surprisingly, we observe that the End-of-Session probability is relatively large in the beginning of a session, most notably in the first interval, but also in the second compared to the third. The reason for this is most likely the large amount of short sessions in the dataset (see Figure 2), such that the model learns that a student is likely to quit relatively fast. These short sessions could have been removed, but we wanted to base the model on the full history of sessions, and not with artificial session gaps.

When the model observes that the session has not ended very early, the End-of-Session probability drops, and a small gradual increase is seen until the 95% interval. In the last interval the End-of-Session probability increases relatively

Sessions consisting of:	AUC
Only homework	0.8228
Partly homework	0.8214
No homework	0.7800

Table 4: The AUC scores of actions in sessions consisting of only homework, partly homework, or no homework actions.

quickly, compared to the gradual increase observed from 10% to 95%. The true End-of-Session action has an average probability of 65.7%. Even though the true End-of-Session action should ideally have a probability closer to 100%, it is still significantly larger than earlier action’s End-of-Session probabilities, which means we are able to create a ranking useful for measuring when a session is likely to end.

4.3 Further analysis

We conduct a further analysis of the model focusing on how the performance varies when considering: 1) sessions with varying degree of homework, 2) sessions with varying length, and 3) students with varying number of sessions.

4.3.1 Homework sessions

As described in Section 2 students are assigned homework through the Edulab system, where, as seen in Table 2, 48.3% of all sessions consist purely of homework. Sessions consisting of partly homework or no homework are at 25.5% and 26.2% respectively. We consider how the model performs in each of the three cases, where the AUC scores can be seen in Table 4.

Sessions consisting of pure homework can be considered fixed in the sense that a teacher has decided their exact content. Thus, one could imagine the general behaviour of a student to be of lesser importance, since a student is just completing the assigned task. However, if a student is likely to quit early, or split the homework into multiple sessions, then previous observed behaviour is still valuable. The AUC score of all sessions in this case is 0.8228, which is marginally larger than the global AUC score of 0.8103, and thus shows this case to be slightly easier to predict. Sessions consisting partly of homework obtain a similar AUC score of 0.8214. This shows that sessions with fewer individual choices are easier to model (unlike sessions without any constraints).

Sessions consisting of no homework, thus only of actions decided by the student, obtain an AUC score of 0.78, thus lower than the global AUC. This follows our intuition and previous observations, since these should be more difficult, as it is entirely up to the student to define their own task and amount of time they spend using the system.

4.3.2 Sessions of varying length

In this section we consider how the model performs on sessions of varying length, where AUC scores across intervals of session lengths can be seen in Table 5. We see that the model is less accurate when modelling very short sessions of length 1-5 (0.6156 AUC), but from lengths 11 and upwards an AUC of 0.80 ± 0.02 is obtained. That the model performs worse on very short sessions is to be expected, since a certain amount of actions are needed to infer the student’s current behaviour. Additionally, we observed in Section 4.2 that

Session length interval	AUC
1-5	0.6156
6-10	0.7604
11-20	0.7859
21-30	0.8200
31-40	0.8060
41-50	0.8333
51-60	0.8109
61-70	0.8088
71-80	0.8157
81-90	0.8103
91-max	0.8016
Divisible by 5	0.8944
Not divisible by 5	0.7568

Table 5: The AUC scores of actions in sessions of varying length, grouped in small intervals. E.g. 6-10 corresponds to actions associated with sessions of length 6 to 10.

the model had a tendency to initially predict large End-of-Session probabilities for all actions, which also explains why the model generally performs poorly for very short sessions.

A large part of the Edulab content is presented by groupings of related questions, e.g. as 5 or 10 quick questions, and the number of assigned homework questions are often divisible by 5 as well. Due to this a large part of the sessions have lengths divisible by 5, and the model should be able to detect and adapt to when these kind of patterns occur. In Table 5 we see that the AUC score of those sessions are 0.8944, which is significantly larger than the global AUC, thus showing that the model is indeed able to detect these patterns. In the case of less structured sessions, i.e. those of lengths not divisible by 5, we obtain a significantly lower AUC of 0.7568. These trends are similar to those observed when considering the amount of homework a session consisted of, and shows that while the model is more accurate when structure is present, it is also able to adapt and provide reasonably high AUC scores in the unstructured case.

4.3.3 Students with varying number of sessions

We consider how the model performs for students with varying system usage, i.e. who has completed a varying number of sessions. In Table 6 we generally see that actions associated with students with more sessions are more accurately predicted, since the intervals of those with more completed sessions obtain larger AUC scores. This shows that the network is able to learn student specific characteristics which are helpful for the task of predicting End-of-Session.

5. CONCLUSION

We presented the problem of determining when a session ends, by modelling the probability of an action being the last in its session, which we denoted as the End-of-Session probability. This problem is difficult to model since we do not get any explicit information about a student’s intent to end a session soon, but only in the action it happens. Additionally, a multitude of reasons for the session ending exist, e.g. by a student finishing an assigned homework task, feeling unmotivated, or when the student has mastered the material. To model this problem we proposed a deep recurrent neural

Student session interval	AUC
1-5	0.7799
6-10	0.7866
11-20	0.7938
21-30	0.8022
31-40	0.8054
41-50	0.8123
51-60	0.8218
61-70	0.8236
71-80	0.8221
81-90	0.8256
91-max	0.8153

Table 6: The AUC scores of actions in sessions of students with varying number of total sessions, grouped in small intervals. E.g. 6-10 corresponds to students with between 6 to 10 completed sessions.

network architecture, that predicts the End-of-Session probability for each student action, by incorporating information from past actions in the current and previous sessions. We consider this a student level model, and for comparison we created a similar session level model, where only the actions associated with a given session were used, and not all previous actions as in the student level model. The student level model obtained an AUC of 0.8103 and the session level model an AUC of 0.5647, thus showing the benefit of learning the student behaviour for this task. Through a detailed error analysis we showed that our model is robust across sessions of different lengths, except for very short sessions of 1 to 5 actions. Similarly, the model gets progressively better for students with more completed sessions, but even with just 1 to 5 sessions an AUC of 0.7799 was obtained. Lastly, the model performed better in sessions with a known structure (e.g. homework), but it was still able to adapt and perform well in sessions where the students chose the content on their own.

In the future we will apply our model, to obtain real-time insights into how the End-of-Session probability progresses throughout sessions and student skill evolution [6]. Additionally, we will combine this problem with Knowledge Tracing, where relatively simple temporal features have already shown to increase performance [16]. In terms of modelling, we will explore more elaborate RNN extensions that have been shown to work well with various sequence-based data [3, 4].

6. ACKNOWLEDGMENTS

The work is partly supported by the Innovation Fund Denmark through the DABAI project (5153-00004A).

7. REFERENCES

- [1] BAYER, J., BYDZOVSKÁ, H., GÉRYK, J., OBSIVAC, T., AND POPELINSKY, L. Predicting drop-out from social behaviour of students. In *EDM* (2012), pp. 103–109.
- [2] HALAWA, S., GREENE, D., AND MITCHELL, J. Dropout prediction in moocs using learner activity features. In *Experiences and best practices in and around MOOCs* (2014), vol. 7, pp. 3–12.
- [3] HANSEN, C., HANSEN, C., ALSTRUP, S., SIMONSEN, J. G., AND LIOMA, C. Modelling sequential music track skips using a multi-rnn approach. In *International Conference on Web Search and Data Mining (WSDM), WSDM Cup* (2019).
- [4] HANSEN, C., HANSEN, C., ALSTRUP, S., SIMONSEN, J. G., AND LIOMA, C. Neural speed reading with structural-jump-lstm. In *7th International Conference on Learning Representations (ICLR)* (2019).
- [5] HANSEN, C., HANSEN, C., HJULER, N., ALSTRUP, S., AND LIOMA, C. Sequence modelling for analysing student interaction with educational systems. In *EDM* (2017), pp. 232–237.
- [6] HANSEN, N. D., LIOMA, C., LARSEN, B., AND ALSTRUP, S. Temporal context for authorship attribution. In *Information Retrieval Facility Conference* (2014), Springer, pp. 22–40.
- [7] HOCHREITER, S., AND SCHMIDHUBER, J. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
- [8] KASSAK, O., KOMPAN, M., AND BIELIKOVA, M. Student behavior in a web-based educational system: Exit intent prediction. In *Engineering Applications of Artificial Intelligence* (2016), vol. 51, Elsevier, pp. 136–149.
- [9] KLINGLER, S., KÄSER, T., SOLENTHALER, B., AND GROSS, M. H. Temporally coherent clustering of student data. In *EDM* (2016), pp. 102–109.
- [10] LIANG, J., LI, C., AND ZHENG, L. Machine learning application in moocs: Dropout prediction. In *International Conference on Computer Science Education (ICCSE)* (2016), pp. 52–57.
- [11] SHEN, S., AND CHI, M. Clustering student sequential trajectories using dynamic time warping. In *EDM* (2017), pp. 266–271.
- [12] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. Dropout: A simple way to prevent neural networks from overfitting. In *The Journal of Machine Learning Research* (2014), vol. 15, JMLR. org, pp. 1929–1958.
- [13] TAX, N., VERENICH, I., LA ROSA, M., AND DUMAS, M. Predictive business process monitoring with lstm neural networks. In *International Conference on Advanced Information Systems Engineering* (2017), Springer, pp. 477–492.
- [14] VASILOUDIS, T., VAHABI, H., KRAVITZ, R., AND RASHKOV, V. Predicting session length in media streaming. In *ACM SIGIR Conference on Research and Development in Information Retrieval* (2017), pp. 977–980.
- [15] XING, W., CHEN, X., STEIN, J., AND MARCINKOWSKI, M. Temporal predication of dropouts in moocs: Reaching the low hanging fruit through stacking generalization. In *Computers in human behavior* (2016), vol. 58, Elsevier, pp. 119–129.
- [16] ZHANG, L., XIONG, X., ZHAO, S., BOTELHO, A., AND HEFFERNAN, N. T. Incorporating rich features into deep knowledge tracing. In *ACM Conference on Learning@ Scale* (2017), pp. 169–172.

Categorizing students' questions using an ensemble hybrid approach

Fatima Harrak
Sorbonne Université
CNRS, Laboratoire
d'Informatique de Paris 6,
LIP6, F-75005 Paris, France
fatima.harrak@lip6.fr

François Bouchet
Sorbonne Université
CNRS, Laboratoire
d'Informatique de Paris 6,
LIP6, F-75005 Paris, France
francois.bouchet@lip6.fr

Vanda Luengo
Sorbonne Université
CNRS, Laboratoire
d'Informatique de Paris 6,
LIP6, F-75005 Paris, France
vanda.luengo@lip6.fr

ABSTRACT

Students' questions categorization is a challenging task as the available corpora are often limited in size (particularly with languages other than English) and require a costly preliminary manual annotation to train the classifiers. Ensemble learning can help improve machine learning results by combining several models, and is particularly efficient to leverage the strengths of very different classifiers. In this paper, we investigate how combining a rule-based annotator (based on keywords identified by an expert) with various machine learning-based approaches and TF-IDF can improve the automated identification of questions asked by 1st year medicine students on an online platform, according to a coding scheme using 4 dimensions. First we evaluated the performance of several models, calculating the kappa between the prediction and the manually labelled dataset, according to each dimension. Then, using a stacking approach, we tried different combinations of them to design a predictive model with a higher performance. The results reveal that the new ensemble models can help to increase the performance for all dimensions of the dataset, in particular those for which the expert rule-based system showed the lowest performance. These results are promising as they indicate that some easy-to-train models can complement more manual approaches, even with a small training set of a few hundreds of annotated questions.

Keywords

Student's question, ensemble learning, stacking, coding scheme, hybrid method, question categorization

1. INTRODUCTION

Categorizing students' questions with limited size of corpora remains a challenging task. The available classification methods require a costly preliminary manual annotation to obtain labeled data, and it is tempting to try training many different classifiers in the hope that combining their predictions would give good performance. One of the most active

areas in machine learning has been in studying methods to build good ensembles of classifiers [3]. The premise that ensembles are often much more accurate than the individual classifiers make them more attractive. Ensemble learning helps improve machine learning by combining several models to obtain an overall classifier which prediction accuracy outperforms every single one of them [7, 9]. Among the existing ensemble approaches, stacking [17] is often used. It consists in training a combining classifier (sometimes called a meta-classifier), in addition to the set of individual classifiers, which takes as input the output of the other classifiers.

In this paper, we used a pre-existing coding scheme to annotate students' questions asked by 1st year medicine students on an online platform, and investigated different approaches to improve the automated identification of their questions. We used the stacking approach by combining heterogeneous classifiers such as a rule-based annotator with various machine learning-based approaches and TF-IDF. Our goal was to answer to two research questions: (RQ1) Can combining different approaches improve the performance of the predictive model? (RQ2) What is the best combination of families of classifiers, and in particular, can a hybrid system (mixing expert rules and machine learning) be relevant?

2. STATE OF THE ART

Annotating a corpus automatically requires using a coding scheme or a taxonomy of sentences, messages or speech acts. Many taxonomies have been used to characterize the types of questions that students ask. Graesser and Person [4] developed a taxonomy of questions asked during tutoring sessions according to the level of thought. Another well-known taxonomy widely used in education, the Bloom's taxonomy [2] and its revisions [1], was originally created in order to help teachers in formulating questions and therefore tends to be more appropriate for teachers' questions (*e.g.* assessment) than for students' ones. The questions coding scheme used here was defined based on the corpus at hand to match them more accurately, even if some overlap exists with categories of existing taxonomies.

Ensemble learning has been investigated in many studies [11, 10] and consists in weighting several individual classifiers, and combining them in order to obtain a classifier that outperforms every single one of them considered separately. First, different classifiers are generated by training models on different features from the training set. The generated

Fatima Harrak, François Bouchet and Vanda Luengo "Categorizing students' questions using an ensemble hybrid approach" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 312 - 317

classifiers are then typically combined by some form of majority or weighted voting. In this work, we do not restrict how the individual classifiers are trained, but we deal with different models and not only probabilistic ones which is a prerequisite for some of these techniques.

The stacking framework introduced by [17], consists in combining multiple classifiers models created by using different learning algorithms on a single dataset. Several variations of this idea have been attempted. Ting and Witten stacked base-level classifiers whose predictions are probability distributions over the set of class values, rather than single class values. The meta-level attributes are thus the probabilities of each of the class values returned by each of the base-level classifiers. Multi-response linear regression, an adaptation of linear regression, is recommended for meta-level learning to predict binary variables [15]. Merz [8] proposed a stacking method called SCANN that uses correspondence analysis to detect correlations between the predictions of base-level classifiers. Todorovski and Dzeroski [16] introduced a new meta-level learning method for combining classifiers with stacking: meta decision trees have base-level classifiers in the leaves, instead of class-value predictions. Researchers in [14] presented a novel bayesian model that relies on combining different models in order to improve the classifier accuracy.

In this paper, we investigated how combining heterogeneous classifiers (derived by different learning algorithms, using different model representations) can help to improve the automated identification of questions using a stacking approach.

3. CONTEXT

The dataset considered in this paper is made of questions asked in 2012 by 1st year medicine/pharmacy students from a major public French university. The Faculty of Medicine and Pharmacy has a specific hybrid training system (reading of the material for the class is done at home, and classroom time is dedicated mostly to Q&A) for their 1st year students. The 1st year is divided into two semesters, each of them ending with a competitive exam (in January and May) on the content studied during the period: only a predefined number of students is allowed to continue in the second year. Between the reading session at home and the classroom session, the students can connect to an online platform to either ask a question, or see questions asked by other students and vote for them if they also want an answer to that question. They cannot however propose an answer to those questions. Then, the questions asked online are sent to teachers to help them prepare their Q&A session. The dataset contains 6457 questions overall asked by 429 students.

4. QUESTION CODING SCHEME

We chose to consider the nature of questions as defined in the coding scheme introduced in [5], in a process involving multiple human coders and several refinement phases. This coding scheme (summarized in Table 1) consists in tagging each question according to 4 independent dimensions: a main one (dimension 1), which is mandatory, and 3 optional ones (dimensions 2 to 4 - cf. Table 1 for the corresponding definition of each dimension). For instance, a question could be a request to re-explain the way something work by providing another example (tagged as Ree (1) on dimension 1,

Table 1: Coding scheme

ID	Dim1	Question type	Description
1	Ree	Re-explain / re-define	Ask for an explanation already done in the course material
2	Dee	Deepen a concept	Broaden a knowledge, clarify an ambiguity or request for a better understanding
3	Ver	Validation / verification	Verify or validate a formulated hypothesis
ID	Dim2	Explanation modality / Quest. subject	Description
1	Exa	Example	Example application (course/exercise)
2	Sch	Schema	Schema application or an explanation about it
3	Cor	Correction	Correction of an exercise in course/exam
ID	Dim3	Explanation type	Description
1	Def	Define	Define a concept or term
2	Man	Manner (how?)	The manner how to proceed
3	Rea	Reason (why?)	Ask for the reason
4	Rol	Roles (utility?)	What's the use/function
5	Lin	Link between concepts	Verify a link between two concepts, define it
ID	Dim4	Verification type (optional)	Description
1	Mis	Mistake / contradiction	Detect mistake/contradiction in course or explanation of teacher
2	Kno	Knowledge in course	Verify knowledge
3	Exp	Expected knowledge	Verify expected information in exam or quiz (assessment)

Exa (1) on dimension 2, Man (2) on dimension 3, and nothing (0) on dimension 4, *i.e.* represented as the dimension vector [1,1,2,0]). It could also be a request to verify (Ver = 3) if a schema (Sch = 2) needs to be known for the final exam (Exp = 3) (which would be tagged as [3, 2, 0, 3]). We considered here a corpus of 923 questions manually annotated according to the 4 dimensions of the coding scheme for training and testing the automated annotators. A subsample of 723 questions was used for training the classifiers, and 200 questions were used to test their performance.

5. AUTOMATED ANNOTATORS

5.1 Approach 1: expert rule-based annotator

We used first a previously developed custom annotator relying on keywords manually identified and associating them a weight [6]. To design it, the human annotator identified from a separate dataset of questions in the corpus the keywords that were indicative of each dimension value (*e.g.* in Dimension1, for the dimension value "Re-explain", some of the keywords identified were "re-explain", "restate", "redefine", "retry", "repeat", "revise", "get back on", *etc.*). For each question, for each dimension, the question was tagged in the dimension according to the value that had the highest number of keywords associated to it (*e.g.* for dimension 1, a question with two keywords associated to the value "re-

Table 2: Kappa between the standalone expert rule-based annotator and the reference manual annotation

Dim1	Dim2	Dim3	Dim4
0.76	0.69	0.70	0.65

explain” and one keyword associated to the value ”validation” would be tagged as a ”re-explain” question).

The automatic annotator is using a set of weights associated to each keyword of each dimension (*e.g.* ”explain”: 7, ”what/how”: 3), and defined using the set of 723 questions. Those weights were determined in two steps: first, the human annotators empirically associated a weight between 1 and 9 to each keyword, depending on whether they thought they were very marginally (1), significantly (5) or very significantly (9) associated to a given dimension. Then in a second step, the automatic annotator was used on the 723 manually annotated questions, and weights were manually adjusted (adding or removing 1) on some keywords for questions for which the manual and automatic annotation were different, iterating until full agreement was obtained on almost all segments from the 723 questions. Finally, the question identified by the values associated to each dimension, is represented as a dimension vector.

The Kappa values per dimension for the annotations coming from human expert and the automatic annotator are given in Table 2.

5.2 Statistical approaches

5.2.1 Data coding: from questions to words vectors

First, we used the French version of WordNet [12], a lexical database linking semantic concepts to each other in an ontology according to a variety of semantic relations (*e.g.* synonyms and hyperonyms). The aim was to transform different synonyms into the same expression in the questions (*e.g.* for dimension value ”Reason” in Dim3, the synonym words ”cause”, ”reason” and ”motif” were replaced in the text by ”why”) to reduce the lexical diversity and consolidate a particular expression for the following treatment. Then, the classical preprocessing steps are used on the corpus of 923 questions: punctuation removal, stemming, tokenization, and stopwords filtering. After extracting the unigrams and bigrams for all questions in the corpus, the weights for the words are computed using two different methods: (1) TF-IDF (described in the next section), (2) counting occurrences (’1’ if the word is in the question, ’0’ otherwise). Each of the 723 questions was represented by a word vector according to (1) or (2). We finally reduced the number of extracted keywords to keep the most important and significant ones using a feature selection technique (removing less frequent and correlated unigrams and bigrams).

5.2.2 Approach 2: TF-IDF

We used TF-IDF [13] to compute term weights. The goal of TF-IDF is to estimate how the words in a given document are representative of that document when compared to a larger set of documents. It combines two complementary metrics: the term frequency (TF), and the invert document

frequency (IDF). TF thus gives a higher weight to the commonly occurring terms and a lower weight to rare terms. The drawback is that some words that are common in a given document but also common in all documents could end up with a weight that is over-representing their real importance. IDF fixes this issue by adjusting the weight with the general importance of the term. Equation 1 describes the method to compute individual TF-IDF weight values for each term (word). We made two different calculation measures of TF-IDF on the corpus of 723 questions.

$$W_{ik} = TF_{ik} \cdot \log \left(\frac{N}{n_k} \right) \quad (1)$$

Where:

W_{ik} = TF-IDF weight for term k in document Q_i

TF_{ik} = frequency of term k in document Q_i

IDF_{ik} = inverse document frequency of term k in document $Q_i = \log \left(\frac{N}{n_k} \right)$

N = total number of documents in the questions corpus

n_k = number of documents in the corpus that contain the term k

The first version consists in calculating four separate TF-IDF on each of the four dimensions, to extract the words that differentiate each category on each dimension. For a given dimension, all the questions manually annotated in each category (*e.g.* ”Re-explain”) were considered as documents (*e.g.* on dimension 1, document1 is the union of questions annotated as ”Ree”). Each document (set of questions) is converted into a corresponding word-weight vector, where each word-weight represents the TF-IDF measure for the word in the document. TF-IDF weight (W_{ik}) was attributed for each term k in document i (i is the number of documents in that dimension, *e.g.* i varying from 1 to 3 for dimension 1). In order to classify new questions, we used the TF-IDF weights calculated on each dimension value from the sample of 723 questions. We attributed TF-IDF weights calculated on the training sample for the corresponding words on the test sample of 200 questions. Then, we chose the simplest ranking function which consists in summing the TF-IDF weights for each question on each dimension value. Therefore, for each question, for each dimension, we tag the question in that dimension according to the value that has the maximum weights. Finally, we calculated the Kappa values between the values found by this model [**TF-IDF+MAX**] for that dimension, and the corresponding values found by the manual annotation (*cf.* first column of Table 3). Two versions were tested: one where the questions were preprocessed using WordNet (*cf.* previous section) and one where they were not. The results obtained were similar in terms of performance, so we decided to keep the version including the preprocessing with WordNet, as it intuitively should generalize better to variations of existing questions.

In the second version, TF-IDF was calculated on the corpus of 723 questions without distinguishing the different dimensions. The questions were not grouped by dimension value, but instead, each question in the corpus was considered as a document (*i.e.* 723 documents overall). The document

Table 3: Kappa between automatic annotation obtained by standalone TF-IDF + different ML methods and the reference manual annotation

Dim.	TFIDF +						
	Max	GLM	GBT	NB	KNN	DT	RI
Dim1	0.66	0.69	0.71	0.47	0.62	0.46	0.61
Dim2	0.39	0.73	0.69	0.12	0.56	0.49	0.36
Dim3	0.66	0.59	0.60	0.43	0.58	0.37	0.52
Dim4	0.58	0.71	0.63	0.37	0.60	0.19	0

is then converted into a corresponding word-weight vector, where each word-weight represents the TF-IDF measure for the word in the question. Finally, we used the word vectors as the input for machine learning techniques to predict the value associated to the question in that dimension (described in section 5.2.3).

5.2.3 Approach 3: ML-based annotator

We tried 6 machine learning (ML) classification techniques (Generalized Linear Model [GLM], Gradient Boosted Trees [GBT], Decision Tree [DT], K Nearest Neighbors [K-NN], Rule Induction [RI] and Naïve Bayes [NB]) for each dimension separately. The appropriate hyper-parameters (such as k for K-NN) were chosen in each case to obtain the highest value and may differ from one table to another. For each classifier, the input was the word vectors and the label to predict was the value associated to the question in that dimension. We considered the corpus of 923 questions as labeled data. Then, we trained the classifiers on the 723 questions and evaluated their performance on an independent sample of 200 questions, to ensure a good estimation of the performance on unseen data. Finally, we calculated the Kappa values between the values found by the classification model for that dimension, and the corresponding values found by the manual annotation. We considered two versions for comparison here as well: the first one using the corpus processed using WordNet, and the second one without the processing with WordNet.

5.3 Results

The kappa values found with the three automated annotators taken individually (expert rule-based, TF-IDF and ML) are provided in Tables 2, 3 and 4 respectively for each dimension. We note that the expert rule-based annotator clearly outperforms both ML-based annotator and TF-IDF only on dimension 1, whereas they almost have similar performances on dimension 3. TF-IDF with the classifier GLM gives the best performance on dimension 4. Furthermore, the ML-based annotation without WordNet performs better than the classifiers using WordNet for all dimensions and particularly on dimension 2.

6. ENSEMBLE HYBRID APPROACH

Our next step consists in building a predictive model with a higher performance to improve the automated identification of questions according to the coding scheme provided in Table 1. Using the aforementioned stacking approach, we tried different combinations of models regardless of which classifier is the best one. Moreover, it does not require any of the classifiers to be probabilistic; they can even be human experts. Our goal was not only to obtain the best classifier

Table 4: Kappa between automatic annotation obtained by standalone different ML methods and the reference manual annotation

Dim.	GLM	GBT	NB	K-NN	DT	RI
Processing using WordNet						
Dim1	0.69	0.70	0.28	0.60	0.73	0.69
Dim2	0.10	0.74	0.10	0.50	0.79	0.37
Dim3	0.68	0.64	0.37	0.61	0.59	0.60
Dim4	0.63	0.66	0.34	0.60	0.48	0.63
Processing without WordNet						
Dim1	0.73	0.69	0.33	0.56	0.74	0.66
Dim2	0.58	0.81	0.12	0.48	0.85	0.29
Dim3	0.70	0.65	0.35	0.60	0.57	0.62
Dim4	0.63	0.67	0.46	0.59	0.10	0.47

performance, but also to do so using a fairly small training set of annotated questions and see if a good performance could be obtained nonetheless.

6.1 Method for stacking

In the first phase, a set of 20 base-level models have been created (1 expert rule-based annotation, 7 TF-IDF annotation and 12 ML-based annotation). In this second phase, we want to train a meta-level classifier that combines the outputs of the base-level models. In other words, we have 20 predictions for each dimension for each of the 200 question segments in the testing set, as well as the 20 manual annotations for these 200 segments that provide a grounded truth, and we want to train a classification model using some subsets of these 20 features. We trained the meta-level classifier using the same aforementioned 6 classification techniques (GBT, GLM, NB, K-NN, DT, and RI) for each dimension separately, using a 10-fold cross validation to ensure a good estimation of the performance (*i.e.* training the models on 180 segments and testing on 20). Finally, for each model we calculated the Kappa values between the values found by that meta-model for that dimension, and the corresponding values found by the manual annotation. Regarding the set of features we considered, we wanted to consider combinations that mixed different set of approaches, and we therefore considered six meta-learning combinations described below. For each of them, the training was performed four times (once for each of the four dimensions - *cf.* Figure 1).

(1) Stacked TF-IDF models: We combined the outputs of the methods using each individual TF-IDF classifier to compute keywords weights (*i.e.* 7 features for each classifier, *cf.* Table 3).

(2) Stacking TF-IDF with expert rule-based annotation: We combined the outputs of the TF-IDF models with the output of the expert rule-based annotator (*i.e.* 8 features for each classifier, *cf.* Tables 3 and 2).

(3) Stacked ML techniques: We combined the outputs of the machine learning-based annotation with the two combinations: processing using WordNet and without it (*i.e.* 12 features for each classifier, *cf.* Table 4).

(4) Stacking ML techniques with expert rule-based annotation: We combined the outputs of the machine learning-based annotation (with and without WordNet) with

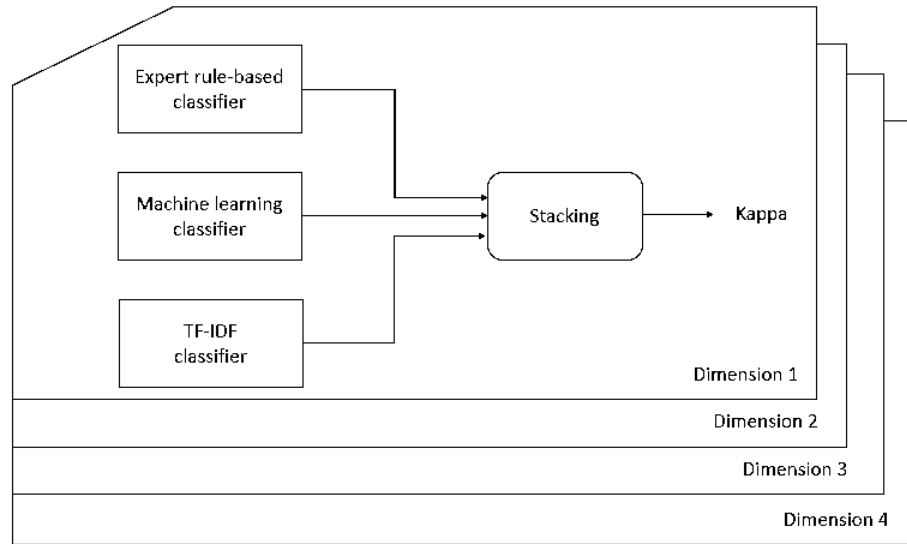


Figure 1: The overall stacking process

the output of the expert rule-based annotation (*i.e.* 13 features for each classifier, *cf.* Tables 4 and 2).

(5) Stacking ML techniques with TF-IDF: We combined the outputs of the machine learning-based annotation (with and without WordNet) with the output of TF-IDF based annotation (*i.e.* 19 features for each classifier, *cf.* Tables 4 and 3).

(6) Stacking ML, TF-IDF and expert rule-based annotation: We combined the outputs of all the existing classifiers: the machine learning-based annotation (with and without WordNet) with the output of TF-IDF and expert rule-based annotation (*i.e.* 20 features for each classifier, *cf.* Tables 4, 3 and 2).

6.2 Results and discussion

The kappa values found with the 6 classification techniques for each dimension are provided in Table 5. Each stacking model was trained individually on each dimension and the highest value obtained for each dimension among the 6 classifiers is tagged in bold, for each set of features considered. For instance, on the first row, we see that when combining the 7 TF-IDF classifiers that predict dimension 1, the best stacking result is obtained with a decision tree (0.75), which outperforms the best individual TF-IDF classifier (0.71 with GBT, *cf.* Table 3). We can notice that Naive Bayes is often the best ensemble classifier among the 6 tested, giving better performance on a small dataset. The best overall performance between the 6 set of comparisons are marked with a star (*): for dimension 1 and 4, it is Naive Bayes combining the ML and the expert rule-based classifiers, for dimension 2 it is Naive Bayes combining TF-IDF and the expert rule-based classifiers, and for dimension 3 it is GBT combining also TF-IDF and the expert rule-based classifiers.

When considering the combinations involving TF-IDF, we see that the combination of several TF-IDF outperforms the

base-level TF-IDF on dimension 1 and 3. The kappa values are overall lower on dimensions 2 and 4, which is probably due to the unbalanced training data in these dimensions (it also explains why sometimes a classifier would obtain a kappa of 0 on these dimensions in the various tables). Moreover, the various TF-IDF classifiers combined with expert-rule based annotator outperforms both the TF-IDF base-level and expert-rule based annotator, as well as the combination of several TF-IDF. Similar results were found for several TF-IDF combined with machine learning, with a slightly better performance than individual classifiers. Overall, if one had to choose only one set of features, the best option is an hybrid ensemble (TF-IDF with expert rule-based annotator), which outperforms on average the model combinations with an average kappa of 0.77 (from the classifiers giving the best performance on each dimension, *i.e.* NB on dimensions 1 and 2, GBT on dimension 3 and K-NN on dimension 4).

When considering the combinations involving ML-based classifiers, the ML-based annotator combined with expert rule-based outperforms slightly the base-level machine learning on dimensions 1, 3 and 4 compared to the other ML combinations. Similarly to TF-IDF, the hybrid ensemble (ML with expert rule-based annotator) gives an average kappa of 0.77 instead of 0.74 for the base-level ML.

The combination of the three types of approaches obtains a performance similar or lower than the two other previously mentioned hybrid ensembles.

7. CONCLUSION

In this paper, we have shown that even with a small training set (less than 1000 questions), it can be useful to add ML-based approaches to complement a manually crafted annotator using a stacking approach to combine classifiers with each other. Using an hybrid ensemble of machine learning-based (or TF-IDF-based) annotators with a previously existing annotator seems to be the best approach, leveraging

Table 5: Kappa values between the ensemble models and the reference manual annotation

Stacked TF-IDF models						
Dim.	GLM	GBT	NB	K-NN	DT	RI
Dim1	0.73	0.74	0.72	0.73	0.75	0.70
Dim2	0	0.35	0.67	0.49	0.51	0
Dim3	0.62	0.70	0.66	0.67	0.68	0.66
Dim4	0.55	0.67	0.68	0.69	0.69	0.67
Stacking TF-IDF + expert rule-based						
Dim.	GLM	GBT	NB	K-NN	DT	RI
Dim1	0.73	0.72	0.76	0.72	0.68	0.71
Dim2	0	0.30	0.80*	0.66	0.48	0
Dim3	0.70	0.79*	0.76	0.77	0.75	0.67
Dim4	0.60	0.66	0.72	0.73	0.67	0.65
Stacked ML models						
Dim.	GLM	GBT	NB	K-NN	DT	RI
Dim1	0.76	0.73	0.80	0.76	0.71	0.68
Dim2	0.30	0.48	0.77	0.59	0.62	0
Dim3	0.62	0.71	0.71	0.72	0.70	0.65
Dim4	0.58	0.65	0.72	0.67	0.68	0.57
Stacking ML + expert rule-based						
Dim.	GLM	GBT	NB	K-NN	DT	RI
Dim1	0.77	0.77	0.80*	0.76	0.70	0.69
Dim2	0.16	0.48	0.77	0.60	0.62	0
Dim3	0.64	0.76	0.71	0.73	0.66	0.64
Dim4	0.60	0.66	0.74*	0.69	0.63	0.59
Stacking ML + TF-IDF						
Dim.	GLM	GBT	NB	K-NN	DT	RI
Dim1	0.77	0.73	0.77	0.76	0.71	0.68
Dim2	0.30	0.52	0.78	0.61	0.62	0
Dim3	0.66	0.75	0.71	0.72	0.70	0.62
Dim4	0.60	0.64	0.71	0.71	0.64	0.61
Stacking ML + TF-IDF + expert rule-based						
Dim.	GLM	GBT	NB	K-NN	DT	RI
Dim1	0.77	0.75	0.78	0.76	0.72	0.68
Dim2	0	0.56	0.78	0.58	0.62	0
Dim3	0.65	0.77	0.72	0.73	0.67	0.61
Dim4	0.61	0.63	0.70	0.69	0.63	0.61

the benefits of each approach. Combining TF-IDF and ML-approaches, however, does not seem as relevant. In our case, the hybrid ensemble models helped in increasing the performance for almost all dimensions, thus replying positively to our two initial research questions. It is worth noting though that the use of WordNet to reduce the vocabulary did not help in increasing the classifiers performance in our case.

One of the limits of this paper is that we considered only a single coding scheme and dataset. The increase in kappas can also be sometimes seen as modest, but this is to be put in perspective with the fact that human coders using this coding scheme rarely can reach a kappa superior to 0.75 on such a task. Moreover, one should note that the dimensions that were improved were the ones that were the furthest from the human coder performance. To conclude, we believe our result can open the perspective to easily improve the performance of various speech act and message annotators which often only rely on expert rules annotators.

8. REFERENCES

- [1] L. W. Anderson, D. R. Krathwohl, P. W. Airasian, K. A. Cruikshank, R. E. Mayer, P. R. Pintrich, J. Rath, and M. C. Wittrock. A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives, abridged

edition. White Plains, NY: Longman, 2001.

- [2] B. S. Bloom and M. B. Engelhart. Furst, E.J. Hill, WH, & Krathwohl, DR *Taxonomy of educational objectives. The classification of educational goals. Handbook I: Cognitive domain*. New York: Longmans Green, 1956.
- [3] T. G. Dietterich. Machine-Learning Research. *AI Magazine*, 18(4):97–97, Dec. 1997.
- [4] A. C. Graesser and N. K. Person. Question asking during tutoring. *American educational research journal*, 31(1):104–137, 1994.
- [5] F. Harrak, F. Bouchet, and V. Luengo. Identifying relationships between students' questions type and their behavior. In *10th International Conference on Educational Data Mining*, pages 402–403, 2017.
- [6] F. Harrak, F. Bouchet, V. Luengo, and P. Gillois. Profiling Students from Their Questions in a Blended Learning Environment. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*, LAK '18, pages 102–110, New York, NY, USA, 2018. ACM.
- [7] Y. Koren. The bellkor solution to the netflix grand prize. *Netflix prize documentation*, 81(2009):1–10, 2009.
- [8] C. J. Merz. Using Correspondence Analysis to Combine Classifiers. *Machine Learning*, 36(1):33–58, July 1999.
- [9] D. Munoz, J. A. Bagnell, and M. Hebert. Stacked hierarchical labeling. In *European Conference on Computer Vision*, pages 57–70. Springer, 2010.
- [10] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*, 6(3):21–45, 2006.
- [11] L. Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, Feb. 2010.
- [12] B. Sagot and D. Fišer. Building a free French wordnet from multilingual resources. In *OntoLex*, 2008.
- [13] G. Salton. Automatic text processing: The transformation, analysis, and retrieval of. *Reading: Addison-Wesley*, 1989.
- [14] E. D. Simpson, M. Venanzi, S. Reece, P. Kohli, J. Guiver, S. J. Roberts, and N. R. Jennings. Language Understanding in the Wild: Combining Crowdsourcing and Machine Learning. In *Proceedings of the 24th International Conference on World Wide Web*, pages 992–1002, Geneva, Switzerland, 2015.
- [15] K. M. Ting and I. H. Witten. Issues in Stacked Generalization. *Journal of Artificial Intelligence Research*, 10:271–289, May 1999.
- [16] L. Todorovski and S. Džeroski. Combining Multiple Models with Meta Decision Trees. In D. A. Zighed, J. Komorowski, and J. Żytkow, editors, *Principles of Data Mining and Knowledge Discovery*, Lecture Notes in Computer Science, pages 54–64. Springer Berlin Heidelberg, 2000.
- [17] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, Jan. 1992.

Identifying Collaborative Learning States Using Unsupervised Machine Learning on Eye-Tracking, Physiological and Motion Sensor Data

Karina Huang
Harvard University
khuang@g.harvard.edu

Tonya Bryant
Harvard University
tonya_bryant@gse.harvard.edu

Bertrand Schneider
Harvard University
bertrand_schneider@gse.harvard.edu

ABSTRACT

With the advent of new data collection techniques, there has been a growing interest in studying co-located groups of students using Multimodal Learning Analytics [3] to automatically identify collaborative learning states. In this paper, we analyze a multimodal dataset (N=84) made of eye-tracking, physiological and motion sensing data. We leverage unsupervised machine learning algorithms to find (un)productive collaborative states. We found a three-states solution where different states (and transitions between them) were significantly correlated with task performance, collaboration quality and learning gains. We interpret these findings in light of collaborative learning theories and discuss their implications for studying groups of students using MMLA.

Keywords

Multimodal Learning Analytics; Eye-tracking; Physiological Sensing; Motion Sensing; Unsupervised Machine Learning.

1. INTRODUCTION

The last decade has seen educational researchers go beyond the study of conceptual learning to understand non-cognitive skills. These skills are considered central for preparing students for the challenges of the 21st century and turning them into resilient, creative, curious and collaborative individuals. Additionally, new learning environments are becoming popular to foster those skills, such as makerspace and digital fabrication labs. While there has been some important progress made in the study of 21st century skills, measuring and assessing them remains a challenge. Most educational researchers and practitioners still rely on traditional collection tools such as participant observations and in-depth interviewing. While research strategies provide valuable insight into learning and development, they are no longer the most efficient way of collecting data.

With the advent of new data collection techniques, however, there has been a growing interest in capturing 21st century skills using Multimodal Learning Analytics (MMLA; [3]). MMLA is about using high frequency sensors, such as eye-trackers, motion sensors, physiological devices and brain sensors to capture students' learning trajectories. Additionally, by combining multiple sensors

it is possible to study collaborative learning groups and capture various aspects of productive collaborations. Traditionally, these sensors have been studied in isolation. The promise of MMLA is to combine multimodal data sources to capture a more holistic picture of students' learning. Being able to capture 21st century skills [6] (such as collaboration) in real time opens new opportunities for providing feedback and designing new kinds of interventions to teach these skills.

The paper is organized as follows. First, we review the literature on several constructs related to collaborative skills that can be captured using high frequency sensors (e.g., Joint Visual Attention, Physiological Synchrony, body postures). We then describe the study that generated our dataset and detail how these constructs were measured. Finally, we present findings where we identified collaborative states using unsupervised machine learning algorithms and discuss their implications.

2. Literature Review

For decades, socio-constructivist theories have emphasized the importance of social interactions for learning (e.g., [12]). Among other things, collaborative learning can help students develop critical thinking skills, increase their motivation, provide a support system and facilitate assessment by making learning visible [10]. Capturing collaborative processes, however, remains a challenge – even though researchers have argued for almost a century that we need more rigorous ways to capture learning processes [23]. In the study of collaborative learning, Dillenbourg [8] argues that “empirical studies have started to focus less on establishing parameters for effective collaboration and more on trying to understand the role which such variables play in mediating interaction. This shift to a more process-oriented account requires new tools for analyzing and modelling interactions”. Below we review how Multimodal Learning Analytics (MMLA; [3]) can help us make a first step in this direction. More specifically, we describe how dual eye-tracking, motion sensors and physiological sensors can provide fine-grained indicators of collaboration.

2.1 Joint Visual Attention and Dual Eye-tracking

Joint Visual Attention (JVA; [4]) is the most fundamental building block by which human beings coordinate their actions, establish a common ground, advance toward a common goal, solve problems, and learn together. It is a construct that encompasses numerous visual processes, and is observed as important for learning to socialize [4], engaging collaboratively [22], and developing social motivation [16] for diverse populations in varying collaborative conditions. The last decade has seen a small but growing number of researchers take advantage of synchronized eye-trackers to quantitatively measure gaze alignment in various collaborative

Karina Huang, Tonya Bryant and Bertrand Schneider "Identifying Collaborative Learning States Using Unsupervised Machine Learning on Eye-Tracking, Physiological and Motion Sensor Data" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 318 - 323

situations interpersonal communication [14]. With the emergence of MMLA, quantifying gaze synchronization in remote learning and problem-solving environments has similarly popularized. In video lectures, projecting the professor's gaze onto the screen (as a substitute for the use of deictic gestures in co-located teaching environments) while making explicit references to information on slides can be useful for students and increase learning gains [20]. In co-located collaborative problem-solving situations, students' level of JVA has been found to be positively correlated to behaviors such as managing group dialogue, reaching consensus, and equally dividing work between members of the group [18]. Regardless of the context, JVA measurement and visualization tools are providing new ways to allow for objective inferences to be drawn about gaze synchronization as it relates to various collaborative states.

2.2. Body Postures and Motion Sensing

Students' use of their bodies has received a great deal of attention from learning scientists over the last decades. Numerous studies have unraveled links between students' understanding of various topics [1, 5] and specific gestures [15]. More generally, there has been a plethora of studies linking people's intuitive representations of everyday situations and bodily language (e.g., embodied cognition [2]). Recently, researchers have started using motion sensors to provide more fine-grained analyses of body postures in collaborative learning settings. For example, [19] found that hand movement could distinguish between students who were more dominant (called "drivers") and those who were more passive (called "passengers").

2.3 Physiological Sensors and Group Synchrony

Researchers have recently started to use EDA sensors (Electrodermal Activity) to look at collaborative learning interactions. [13] describes four measures of physiological synchrony in small groups of students: Signal Matching (SM), Instantaneous Derivative Matching (IDM), Directional Agreement (DA), and Pearson's correlation coefficient (PC). They found that IDM was related to collaboration quality and task performance, and DA with learning gains. In a separate publication, we applied the same methodology to the dataset described in this paper and found that those indices provided significant predictors for collaborative learning [7]. DA was significantly correlated with collaboration quality, IDM with task performance, and PC with learning gains. In a different study, [9] found that DA was the best predictor for task performance. It is interesting to note the discrepancy between the findings above, which is likely due to the nature of the task and the way we operationalized our constructs. But overall, researchers have found that physiological synchrony seems to be sensitive to social interactions in a variety of contexts.

In summary, there is significant evidence that collaborative learning processes can be captured using multimodal sensors. This paper goes one step further by combining modalities together, instead of studying them in isolation.

3. METHODS

3.1. The Study

The dataset used in this paper was collected as a part of a Multi-Modal Learning Analytics study [21]. 84 participants (Male = 40%; Female = 60%) with no prior programming experiences were randomly assigned to dyads ($N_{\text{dyad}} = 42$) and programmed a robot to solve a series of mazes during 30-minute sessions (Fig. 1). Each dyad was randomly assigned to one, both, or neither of two

designed interventions: 1) a verbal explanation of the benefits of collaboration (e.g. past research findings using equity of speech time as indication of collaboration quality), and 2) real-time visualizations showing relative verbal contributions of each participant. The number of dyads was evenly distributed among four experimental conditions. For the analyses reported below, we analyze the aggregated data and did not consider the four experimental conditions.

During each session, we used two Empatica E4 wrist sensors to track participants' physiological activities, two Tobii Pro Glasses 2 eye-trackers to capture eye gaze, and one Kinect sensor to record movement as well as facial expressions. Participants were given a survey before and after the study for assessment of their computational thinking and collaboration experiences. A dyad's collaboration, task performance, and learning outcomes were assessed by the researcher responsible for running the session; ratings were given on nine scales based on prior work by Meier, Spada, and Rummel [11] (inter-rater reliability of 0.65 – i.e., 75% agreement). Analyses of learning gains, coding schemes, inter-judge reliability scores and individual sensor data have been reported in [21]. The current analysis aimed to combine all sensor data in order to identify collaborative states.

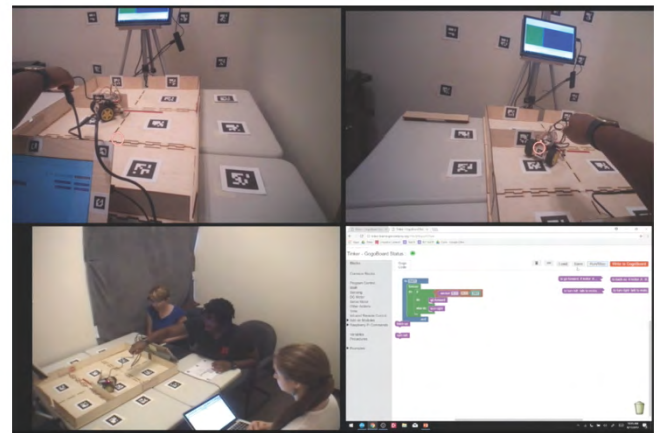


Figure 1. Two participants from the study. The top images show the video feed from the mobile eye-trackers (with a participant's gaze shown on the top right image). The bottom left image shows a 3rd person perspective. The bottom right image shows the programming environment.

3.2. Data Collection

3.2.1. Empatica

The Empatica E4 wristbands collected participants' accelerometer, blood volume pulse (BVP), interbeat intervals (IBI), electrodermal activities (EDA), and heart variability (HR). The current study focused on the EDA data, which is a measure of skin potential, resistance, conductance, admittance and impedance. Participants were asked to tag the wristbands before and after each step during the sessions (i.e. before and after completing the maze task). We synchronized the dyadic data according to the tags and timestamps of the sessions. The resulting data frame per dyad contained timestamp, four aggregated measures of physiological synchrony described in the Measures section below, and EDA values of each participant. Five dyads were removed from the current analysis because the sensor data was missing, too noisy, or identified as outliers (see [21]).

3.2.2. Tobii Pro Glasses 2

The Tobii eye-trackers generated data at 50Hz per second and recorded the x and y coordinates of each participant's eye-gaze relative to its point of view. The resulting data frame per dyad contained time indicated by second (ranging from roughly 1s to 1800s), the x and y coordinates of each participant's eye gaze and counts of joint visual attention (JVA) by pixel distance. The eye-tracking data was synchronized with the EDA data by briefly presenting a fiducial marker on the computer screen between each step of the session; participants were asked to tag this event on their wristband as accurately as possible. While we were able to clean and synchronize most dyadic data by seconds, two groups were excluded from the current study due to missing data.

3.2.3. Kinect

The Kinect motion sensor captured around 100 variables related to a participant's body joints and skeleton. The sensor generated data at 30 Hz per second, resulting in about 3,000 observations per second per participant. Noisy data (e.g. when the session facilitator entered the Kinect frame) were removed for each group, after which data were aggregated by second according to timestamps generated by the Kinect sensor. Researchers manually trimmed the Kinect data for each 30-minute session based on video records and aligned them with the eye-tracking and EDA data. Nine dyads were removed from the current study due to missing data.

3.2.4. Synchronizing All Data

The Empatica and Kinect data were synchronized by trimming each session to exactly 30 minutes and outer-joining sessions' data on the timestamp column. Per-dyad eye-tracking data were synchronized by matching the "second" column (i.e. from 1s to 1800s during a 30-minute session; see Fig. 2) generated based on timestamps of the EDA data. For analysis purpose, we concatenated all per-dyad data into a master data frame, with an additional column indicating to which session the data belonged to. Due to an unequal amount of data loss between sensors, two datasets were created for analysis: 1) Combined EDA and JVA ($N_{\text{dyad}} = 35$), and 2) Combined EDA, JVA and Kinect ($N_{\text{dyad}} = 31$). The current analysis used the second dataset, including 67,656 rows and 19 columns of by-second original and scaled data from all sessions investigated (Fig. 2).

session	second		DA	PC	IDM	SM	jva100	moveDiff	headDiff	shoulderDiff
0	2	1	0.233333	-0.447563	0.177397	0.208709	24.0	1.941329	0.047057	0.658437
1	2	2	0.232708	-0.445837	0.177548	0.207804	7.0	2.013969	0.040868	0.654975
2	2	3	0.233125	-0.442000	0.177468	0.206289	18.0	1.825878	0.048828	0.642805
3	2	4	0.234583	-0.439776	0.176532	0.205172	23.0	1.675585	0.037022	0.648898
4	2	5	0.234375	-0.438995	0.175948	0.204015	6.0	1.967405	0.042931	0.646596

Figure 2. A snapshot of the final data frame. Note that the scaled column measures are excluded for legible visualization.

3.3. Data Processing

3.3.1. Electrodermal Activities (EDA)

Four measures of physiological synchrony were computed based on participants' electrodermal activities (refer to [7] for an exhaustive description of these measures and related analyses): 1) Pearson's Correlation (PC) represented the linear relationship between the EDA level of each participant in a dyad; a strong, positive correlation indicated that the dyad was physiologically activated at similar times. 2) Directional Agreement (DA) captured whether the EDA level of each participant in a dyad increased or decreased at the same time steps; an increase in DA value in the positive direction indicated higher physiological synchrony. 3) Signal Matching (SM) was computed as the area between data curves of each dyad. A greater SM value indicated lower

physiological synchrony. 4) Instantaneous Derivative Matching (IDM) computes, for each dyad, the level of signal matching between slopes of participants' signal curves. A higher IDM value indicated lower physiological synchrony between participants.

3.3.2. Joint Visual Attention (JVA)

JVA was qualified by looking at participants' location of eye gaze after mapping these coordinates into a common plane (Fig. 3; see [17] for the complete procedure of computing JVA). The current analysis looked at the number of JVA per second where a dyad's eye gazes were within 100 pixels of each other.

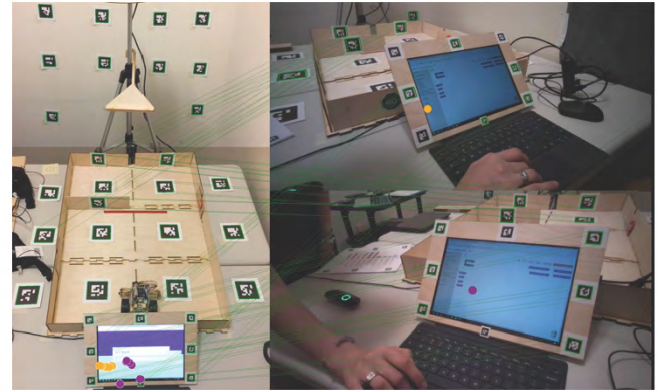


Figure 3. The procedure used to compute Joint Visual Attention (the left side shows the data from the mobile eye-trackers, and the right side shows how the participants' gaze were mapped onto a common plane using a homography).

3.3.3. Kinect

We explored various collaborative measures based on prior work [18, 23]. The current analysis aggregated three measures of movement differences per dyad: 1) Total difference in movement (MoveDiff), 2) Vertical difference in head orientation (HeadDiff), and 3) Horizontal difference in shoulder orientation (ShoulderDiff). Total movement was computed by taking the Euclidean distance of all joint coordinates; difference in movement within dyad was the absolute difference in the participants' total movements. Vertical difference in head orientation was the absolute difference in the y coordinates of participants' heads. Horizontal difference in shoulder orientation was calculated by taking the absolute value of the difference between 1) the absolute difference in x coordinate of the left shoulder of the left participant and the x coordinate of the right shoulder of the right participant, and 2) the absolute difference in the x coordinate of the right shoulder of the left participant and the x coordinate of the left shoulder of the right participant.

3.3.4. Outcomes Measures

The study generated three types of outcome measures: collaboration quality [11] (sustaining mutual understanding, dialogue management, information pooling, reaching consensus, task division, time management, technical coordination, reciprocal interaction, individual task orientation, and Collaboration – the sum of those scores), overall task performance (task performance, task understanding, improvement over time) and learning gains. Collaboration and Task performance of each dyad was hand-coded by the experimenter at the end of the session. The dyad's learning gain was assessed through a pre-test and post-test. For more information, please refer to [21].

3.4. Analysis Strategy

We used K-Means Clustering with Euclidean distance to identify different collaborative states. In particular we attempted clustering using all sensor data simultaneously. All data were transformed into

z-scores before clustering; the scaled values reported in results refer to the z-scores. Note that the clustering assignment was performed on the aggregated data with per-second data from all dyads. The collaborative states were identified regardless of group and time. We used the elbow curve to identify the optimal number of clusters for each clustering strategy; the current analysis used within-cluster sum of squared as the indication of distortion. We proceeded our analysis using $K = 3$.

Upon assigning per-second data to clusters, we computed 1) time spent in each cluster, and 2) transition probabilities between clusters for each session. Correlations between time in cluster, transition probabilities, and each qualitative outcome measure were then investigated and visualized. The results section below summarizes our findings by outcome measures.

4. RESULTS

4.1. Correlation Check

To check for underlying relationships between our sensor data aggregated at the second level and the qualitative outcomes, we first checked for correlations between each sensor and qualitative measure. Significant correlations were observed between SM and Learning ($r = -0.4$, $p = 0.025$), and between JVA and sustaining mutual understanding ($r = 0.41$, $p = 0.027$). In accordance with previous analysis [7], no other significant correlation was observed.

4.2. Cluster Centroids

Centroid 1 values were the highest in all movement variables, suggesting cluster 1 as a state where dyads exhibited the most total movement difference, vertical difference in head orientation (e.g. a person standing up versus the other seated), and horizontal difference in shoulder orientation (when dyads were far apart from each other). Centroid 2 values were the highest in DA, PC, JVA, and the lowest in IDM, and SM; cluster 2 indicated a state where dyads were physiologically synchronized and actively sharing eye gaze. In contrast, cluster 3 appeared to be a state where dyads were the most desynchronized. Centroid 3 had the highest SM, IDM, and the lowest DA, PC values. Table 1 provides a summary of the 3 clusters identified by K-means Clustering, we will use the identified states to address the clusters in the following sections.

Cluster	Physiological Synchrony	Joint Visual Attention	Movement Difference	State
1	Average	Average	Highest	Neutral
2	Highest	Highest	Low	Collaborative
3	Lowest	Lowest	Low	Non-Collaborative

Table 1. Summary of clusters by sensor data.

4.3. Collaboration

Figure 5 represents the scaled and unscaled sensor data values by cluster centroid. The overall collaboration measure ($r = 0.50$, $p = 0.009$) was significantly correlated with time spent in the collaborative state. Time spent in the collaborative state was significantly correlated with sustaining mutual understanding ($r = 0.42$, $p = 0.031$), dialogue management ($r = 0.51$, $p = 0.008$), reaching consensus ($r = 0.48$, $p = 0.013$), task division ($r = 0.49$, $p = 0.012$), and reciprocal interaction ($r = 0.47$, $p = 0.015$). By interpretation of the EDA measure, dyads were highly, physiologically synchronized in the collaborative state. In contrast, dyads were highly desynchronized in the non-collaborative state,

as the state corresponds to the lowest DA, PC and highest SM, IDM values by clustering. Time spent in the non-collaborative state was significantly, negatively correlated with dialogue management ($r = -0.49$, $p = 0.008$), task division ($r = -0.45$, $p = 0.015$) and collaboration ($r = -0.42$, $p = 0.030$).

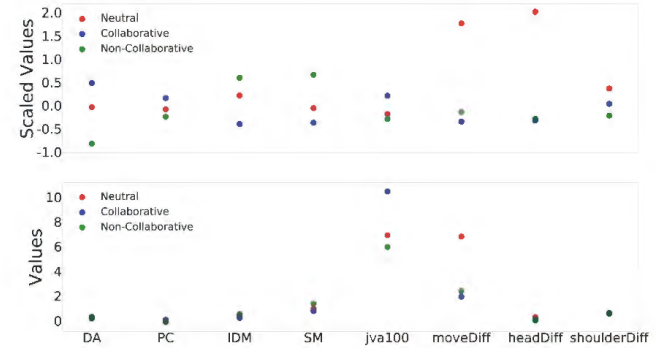


Figure 4. Sensor data mean values by cluster.

Correlations between state transition probabilities and qualitative outcomes rendered the same interpretation: the higher the probability that a dyad would transition into a desynchronized, or non-collaborative state, the lower the rating in collaboration ($r = -0.41$, $p = 0.027$).

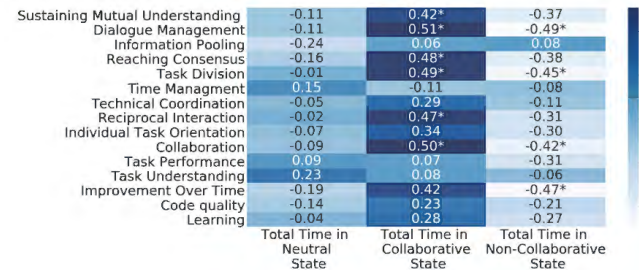


Figure 5. Correlations (left) and p-values (right) between time spent in cluster and qualitative outcome variables (* $p < 0.05$).

4.4. Task Performance

As shown in Fig. 6, time spent in the non-collaborative state was significantly correlated with improvement over time ($r = -0.47$, $p = 0.018$). The more likely a dyad were to transition into the desynchronized state, the lower the rating for task understanding ($r = -0.47$, $p = 0.01$, See Figure 6), and improvement over time ($r = -0.53$, $p = 0.005$). Furthermore, there was a marginally significant correlation between the probability of remaining in the neutral state and code quality ($r = -0.37$, $p = 0.042$); the more a dyad was different in movement, the lower the code quality as evaluated by the session facilitator.

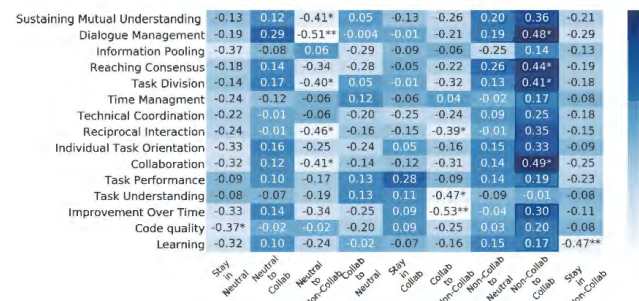


Figure 6. Correlations (top) and p-values (down) between transition probabilities and qualitative outcome variables (* $p < 0.05$, ** $p < 0.01$).

4.5. Learning Gain

We did not observe any significant correlations between learning and time spent in any of the collaborative state identified. However, learning was significantly and negatively correlated with the probability of remaining in the non-collaborative state ($r = -0.47, p = 0.008$). The more likely a dyad were to stay desynchronized, the lower the rating in learning gain.

5. DISCUSSION

Overall, our results suggest that K-Means Clustering is an effective method for identifying collaborative states. In accordance with previous findings, higher JVA in the current analysis significantly correlated with higher ratings of collaboration, more specifically in sustaining mutual understanding. This implies that sharing gaze facilitates collaboration by making aware the object, or the intent, of communication. In comparison with previous correlation findings [7], we were able to draw connections between the EDA measures and collaborative outcomes using clustering analysis. Specifically, physiological synchrony within dyads correlated with higher ratings in quality of collaboration, including sustaining mutual understanding, dialogue management, reaching consensus, and task division. This indicates, intuitively, that when participants in a dyad were in sync with each other, they were more likely to agree, understand, and coordinate in task with each other. Moreover, the larger the difference in movement and position within a dyad, the lower the code quality. One interpretation could be that the longer a dyad spent apart from each other, the less collaborative they were, or were deemed to be, and therefore, the lower the resulting code quality.

In a case study that compared the most collaborative (Group 11) and most non-collaborative (Group 5) groups, we observed that desirable collaborative qualities support the narrative, and the disaggregated graphs (Fig. 7) we created to depict them, that collaborative states are closely associated with levels of JVA and DA value.

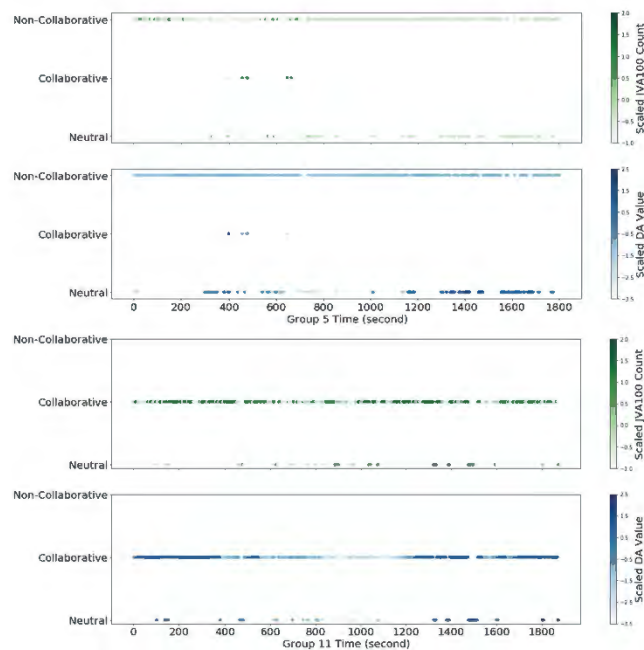


Figure 7. Progression of collaborative states by DA and JVA value in group 5 and group 11.

Though Groups 11 and 5 displayed collaborative (Group 11) and non-collaborative (Group 5) qualities rather consistently across the activity, each group mirrored qualities of the other. For example, while thinking aloud was a consistent quality exhibited by Group 11 across the activity, the participant on the left almost always remained in an observational role, which could have led to low learning gains. However, Group 11 achieved the highest learning gains of all groups in the study. One reason for this outcome could be the intent of the observer. High learning gains and consistent observation seemed to be a mode of learning for the participant on the left. This explanation supports an interpretation of observation not as a culprit of poor collaboration, but an assistant to learning, given a particular learning context. On the other hand, Group 5 also showed behaviors that were contrary to main themes that arose from their interactions. Take for instance their dialog that occurred in the middle of the activity. There, they exhibited seemingly desirable qualities in collaborative learning such as asking for help, asking clarifying questions, using demonstrative actions such as re-running code to convey conception of problem, and a continuous interactive dialog; however, demonstration of these qualities was more of an abnormality than a change in behavior. Furthermore, each participant steadfastly performed their roles. One interpretation of why this temporary change in mode of operation did not stick goes back to the very definition of collaboration: a result of continuous attempts to construct and maintain a shared problem space. This means that establishing a collaborative state two thirds of the way through the activity is perhaps too difficult of a cognitive shift when working modes have been established.

Main Results	Interpretation	Implication
Time spent in cluster 2 is sig. correlated with collaboration quality	Productive collaboration is characterized by increased levels of JVA and physiological synchrony	combining JVA values with PCIs provide better predictors for collaboration quality than just individual JVA or PCI
Staying in cluster 1 or 3 is neg. correlated with learning	The more participants stay in a state where they are not looking at the same place and are physiologically desynchronized prevents them from learning	we could potentially detect when people are in this state in real-time and offer suggestions for getting participants back on track.
Time spent in cluster 3 is neg. correlated with task performance	Non-collaborative states are associated with less task understanding and improvement over time.	*we didn't observe a positive significance with cluster 1, perhaps that while being highly synchronized and actively sharing eye gaze doesn't necessarily leads to more task understanding / improvement overtime; there's a reverse effect when participants were not at all collaborative by EDA/JVA measures.
Staying in cluster 1 is neg. correlated with code quality	The more time participants spent apart from one another, the lower the code quality.	Spending time apart from one another may indicate individual exploration, which in learning settings, may imply that participants were in need of guidance in order to proceed with task.

Table 2. Summary of results.

Finally, we found learning gains to be significantly associated with EDA measures. Particularly, the more likely a dyad were to remain desynchronized physiologically, the less likely that they had learned, or were evaluated to have learned from the task. Overall, we found that combining multimodal measures of collaboration together (e.g., eye-tracking, physiological, motion data) provides us with richer results: we found more (and stronger) significant correlations with our dependent measures. Table 2 summarizes the main results of this paper.

Nonetheless, it is important to note the limitations of the current analysis. For one, the aggregated Kinect measures utilized in the

current analysis might not have best captured motor differences between collaborative and non-collaborative dyads. The current analysis only examined dyad's motor differences on single dimensions, future work should aggregate movement measures based on multiple dimensions (e.g. movement angle) in order to better capture (non-)collaboration in motion. As we identified the number of collaborative states ($K = 3$) using distortions computed with the current dataset, it is possible that this number is not definitive and is unique to the current study. We concluded from exploratory analysis that the implications differed as we increased the number of clusters, and/or reduced our variable dimensions. Moving forward, it is of our interest to find the optimal combination of measures, and the optimal number of states that best characterize the (un)productive collaboration.

5. CONCLUSION

The current study used unsupervised machine learning algorithms to effectively identify different states of collaboration. Combining eye-tracking and physiological-activity data better predicted collaboration quality than the two types of sensor data apart. The longer two partners were not sharing gaze, and were desynchronized from one another, the worse their task performance and the less they learned. Identification of collaborative states and their characteristics through sensor data potentially allows us to monitor collaboration in real-time, detect ineffective cooperation, and keep partnership intact. Future work should explore movement measures of various dimensions to best capture participants' postures and motions.

In summary, this paper contributes to the application of MMLA in open-ended learning environments for capturing 21st century skills. We argue that multimodal sensors can capture different aspect of productive collaboration, and that combining them can provide us with a more complete picture of productive social interactions. Because we tend to "teach what we can measure", developing tools that can capture 21st century skills is a crucial step toward studying and fostering them. This paper makes a first step in this direction by leveraging multimodal sensor data.

6. REFERENCES

- [1] Abrahamson, D. 2009. Embodied design: constructing means for constructing meaning. *Educational Studies in Mathematics*. 70, 1 (Jan. 2009), 27–47.
- [2] Anderson, M.L. 2003. Embodied cognition: A field guide. *Artificial intelligence*. 149, 1 (2003), 91–130.
- [3] Blikstein, P. and Worsley, M. 2016. Multimodal Learning Analytics and Education Data Mining: using computational technologies to measure complex learning tasks. *Journal of Learning Analytics*. 3, 2 (2016), 220–238.
- [4] Bruinsma, Y. et al. 2004. Joint attention and children with autism: A review of the literature. *Mental Retardation and Developmental Disabilities Research Reviews*. 10, 3 (2004), 169–175.
- [5] Church, R.B. 1999. Using Gesture and Speech to Capture Transitions in Learning. *Cognitive Development*. 14, 2 (Apr. 1999), 313–342.
- [6] Dede, C. 2010. Comparing frameworks for 21st century skills. *21st century skills: Rethinking how students learn*. 20, (2010), 51–76.
- [7] Dich, Y., Reilly, J. & Schneider, B. 2018. Using Physiological Synchrony as an Indicator of Collaboration Quality, Task Performance and Learning. In *International Conference on Artificial Intelligence in Education* (pp. 98–110). Springer.
- [8] Dillenbourg, P. et al. 1996. The evolution of research on collaborative learning. *Learning in Humans and Machine: Towards an interdisciplinary learning science*. 189–211.
- [9] Elkins, A.N. et al. 2009. Physiological compliance and team performance. *Applied ergonomics*. 40, 6 (2009), 997–1003.
- [10] Laal, M. and Ghodsi, S.M. 2012. Benefits of collaborative learning. *Procedia - Social and Behavioral Sciences*. 31, (Jan. 2012), 486–490.
- [11] Meier, A. et al. 2007. A rating scheme for assessing the quality of computer-supported collaboration processes. *International Journal of Computer-Supported Collaborative Learning*. 2, 1 (Feb. 2007), 63–86.
- [12] Palincsar, A.S. 1998. Social constructivist perspectives on teaching and learning. *Annual review of psychology*. 49, (1998), 345.
- [13] Pijeira-Díaz, H.J. et al. 2016. Investigating collaborative learning success with physiological coupling indices based on electrodermal activity. (2016), 64–73.
- [14] van Rheden, V. et al. 2017. LaserViz: Shared Gaze in the Co-Located Physical World. *Proceedings of the Eleventh International Conference on Tangible, Embedded, and Embodied Interaction* (New York, USA, 2017), 191–196.
- [15] Roth, W.-M. 2001. Gestures: Their Role in Teaching and Learning. *Review of Educational Research*. 71, 3 (Sep. 2001), 365–392.
- [16] Salley, B. and Colombo, J. 2016. Conceptualizing Social Attention in Developmental Research. *Social Development*. 25, 4 (2016), 687–703.
- [17] Schneider, B. (accepted). Unpacking Collaborative Learning Processes during Hands-on Activities using Mobile Eye-Tracking. In *the 13th International Conference on Computer Supported Collaborative Learning*.
- [18] Schneider, B. et al. 2016. Using Mobile Eye-Trackers to Unpack the Perceptual Benefits of a Tangible User Interface for Collaborative Learning. *ACM Transactions on Computer-Human Interaction*. 23, 6 (Dec. 2016), 1–23.
- [19] Schneider, B. and Blikstein, P. 2015. Unraveling students' interaction around a tangible interface using multimodal learning analytics. *Journal of Educational Data Mining*. 7, 3 (2015), 89–116.
- [20] Sharma, K. et al. 2016. Visual Augmentation of Deictic Gestures in MOOC Videos. (Jul. 2016).
- [21] Starr, E.L., Reilly, J.M. and Schneider, B., 2018. Toward Using Multi-Modal Learning Analytics to Support and Measure Collaboration in Co-Located Dyads. In *Proceedings of the 13th International Conference of the Learning Sciences*, London, UK, p. 448–455.
- [22] Tomasello, M. et al. 2005. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and Brain Sciences*. 28, 5 (Oct. 2005), 675–691.
- [23] Werner, H. 1937. Process and achievement—a basic problem of education and developmental psychology. *Harvard Educational Review*. (1937).

Generalizability of Sensor-Free Affect Detection Models in a Longitudinal Dataset of Tens of Thousands of Students

Emily Jensen
University of Colorado Boulder
emily.jensen@colorado.edu

Stephen Hutt
University of Colorado Boulder
stephen.hutt@colorado.edu

Sidney K. D'Mello
University of Colorado Boulder
sidney.dmello@colorado.edu

ABSTRACT

Recent work in predictive modeling has called for increased scrutiny of how models generalize between different populations within the training data. Using interaction data from 69,174 students who used an online mathematics platform over an entire school year, we trained a sensor-free affect detection model and studied its generalizability to clusters of students based on typical platform use and demographic features. We show that models trained on one group perform similarly well when tested on the other groups, although there was a small advantage obtained by training individual subpopulation models compared to a general (all-population) model. Lastly, we perform a series of simulations to show how generalizability is affected by sample size. These results agree with our initial analysis that individual subpopulation models yield a small advantage over all-population models. Additionally, we show that training sizes smaller than 1,500 yield unstable models which make generalizability difficult to interpret. We discuss applications of this work in the context of developing large-scale affect detection models for diverse populations.

Keywords

Affect Detection, Clustering, Generalizability, Sensor-free, Online Learning

1. INTRODUCTION

Computer-enabled classrooms and online learning environments are becoming increasingly common methods of learning [12, 25]. Compared to traditional classroom settings, students must be more self-regulated when interacting with online platforms [15]. In [20], Pekrun discusses how emotion and its regulation are key factors in educational achievement. It is then important to consider student affect when developing intelligent tutors and educational platforms. A review of affect-sensitive instructional strategies, particularly for intelligent tutors [5], discusses how affect- and motivation-sensitive strategies can promote student engagement. However, the authors found that a “one-size-fits-all approach, where variants of the same strategy are indiscriminately used for all learners and in all situations” limits the overall effectiveness of these tutors in targeting individual student needs. This observation motivates a more detailed analysis of how affect detectors trained on a general population generalize across different subpopulations.

In this work, we extend previous research exploring the generalizability of sensor-free affect detectors. We trained models

predicting positive and negative affective states using interaction data from an online algebra learning platform along with self-reported affect. A novel component of our work compared to previous work (e.g., [2]) is the scope of our dataset, which encompasses 69,714 students across a nine-month period. We extend the previous work in [9]. This enables a more detailed exploration of generalizability than was previously achievable.

1.1 Related Work

Reviews of issues and methods of sensor-free affect detection are covered in other work, which we summarize here. Baker and Ocuppaugh review work in sensor-free affect detection in educational software and discuss methods for collecting ground-truth labels [3]. Specific to this work, they note that student-generated responses are likely more accurate than labels from external coders. Second, [9] reviews a representative collection of sensor-free affect detection models developed in authentic classroom environments. The authors conclude that the studies show the potential success for sensor-free affect detection models in authentic environments but are limited by small sample sizes (20-646 students) from mostly homogenous samples, which limits claims or tests of generalizability.

Recent work in machine learning and prediction calls for increased awareness of how models perform for individual subpopulations in addition to overall accuracy. In [10], Kusner et al. introduce counterfactual fairness, where models should be unaware of protected attributes such as gender and race. Fair models should generalize by generating similar predictions for individuals with similar features, regardless of their protected attributes. In [26], Sculley et al. suggest slicing analysis as a method to evaluate fairness, where predictive model performance is evaluated by “slicing” along subpopulations or protected attributes. This is an alternative to measuring overall model accuracy, which can ignore disadvantaged subpopulations. In response, Gardner et al. present a framework for using slicing analysis in predictive modeling [7].

Related to this discussion on generalizability, several studies have measured how models generalize across cultural contexts. Ogan et al. [18] found differences in collaboration, engagement, and student needs between cultural groups. In [24], San Pedro et al. trained models detecting student carelessness in Philippines. They showed generalizability by testing these models on previously collected data from students in the USA. In [27], Soriano et al. compared models of help-seeking behavior. By training models on each group and testing on the other groups, they showed that models for Philippines and USA generalize to each other but not to Costa Rica.

Besides cross-country generalization, several studies investigated how predictive models generalize over demographic attributes. In [17], Ocuppaugh et al. trained affect detection models on rural, suburban, and urban students. By training models on each group and testing on the other groups, they found that models for urban and suburban students generalized to each other but not to rural students. In [23], Samei et al. trained models of teacher question

Emily Jensen, Stephen Hutt and Sidney K. D'Mello
"Generalizability of Sensor-Free Affect Detection Models in a Longitudinal Dataset of Tens of Thousands of Students" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 324 - 329

asking behavior using data from urban and non-urban classrooms. They showed generalizability using the methods from [17].

Other studies measured the generalizability of predictive models over time. In [1], Baker et al. trained models detecting gaming the system behavior in a cognitive tutor. They showed generalizability by training models on data from three sessions and testing on the remaining session. In [4], Bosch et al. trained face-based affect detection models. They showed generalizability by training models on data from one day and testing on the other day.

Finally, some studies measured generalizability between different tasks or subjects. In [28], Stewart et al. compared models of mind wandering trained on students reading a scientific text or watching a narrative film. They found models trained on the narrative film dataset generalized to the scientific text dataset, but models trained on the scientific text dataset only generalized to the narrative film dataset after adjusting the predicted mind wandering rate. In [9], Hutt et al. found that models trained on data from students enrolled in Algebra 1 generalized to students enrolled in Geometry using “generic activity features” specifically designed for generalization.

1.2 Contribution of Current Study

This work contributes to the field of generalizability in sensor-free affect detection in three important ways. First, we extend beyond previous work by using data from a large, heterogeneous sample of students. Besides the noted studies that compare country-wide cultural differences, previous work relies on homogeneous samples such as individual schools, which yield sample sizes of hundreds of students. As discussed in [2], these sample sizes do not allow researchers to draw conclusions about the studied categories as a whole, so generalizability can only be tested in a minimal sense. In this study, we collected affect data from 69,174 students at 1,898 schools in the state of Florida. Because Florida closely represents the demographic composition of the United States in terms of race and ethnicity [29, 30], this allows us to study the generalizability of our models to other students in the country.

Second, we measure the generalizability of our models in terms of usage characteristics over an entire school year. In previous studies, data are collected during one or a few sessions, which overlooks long-term student behavior. This work uses interaction logs from an entire school year and measures student use over several sessions. We use clustering analysis to identify common usage patterns and show that our models generalize across these clusters.

Lastly, we provide simulation experiments to inform the number of instances needed in order to construct generalizable models. Specifically, we estimate the advantage obtained by training models on individual groups across different sample sizes.

2. DATA

We used a previously published dataset [9] but all analyses reported here are new.

2.1 Algebra Nation

Data was collected through Algebra Nation, an online math learning platform developed by Study Edge. Algebra Nation supports over 150,000 students studying Algebra 1, Algebra 2, and Geometry each semester. Students can use Algebra Nation in a variety of contexts; some teachers integrate the platform into their regular classroom time while some students only use it to study or help with homework. Students can access Algebra Nation using a mobile app or on the internet (<https://www.algebranation.com/>). For this study, we used data from students enrolled in Algebra 1.

In Algebra Nation, course material is organized according to state mathematics standards. Although the topics are ordered according

to the curriculum, students are free to skip topics as necessary or learn the material in a different order.

For each topic, students can watch a video lecture from one of several tutors. In addition to watching videos, students can use the *Test Yourself* quiz feature for each topic, which randomly selects 10 questions aligned with state standards. After attempting a quiz, students can review feedback on their answers or watch solution videos. Lastly, students can get more help through the *Discussion Wall* where they can interact with other students and study experts hired by Algebra Nation. Students can earn *karma points* by answering questions posted by other students. However, students primarily spend time watching videos and taking quizzes rather than engaging in the social functions of the platform.

2.2 Affect Surveys

Due to the large number of students in the study and because students can use the platform in multiple contexts, we collected ground-truth affect labels using a self-report survey rather than through expert coders or human observers (see [16, 22]). These surveys were pseudo-randomly triggered based on student activity on the platform. Specifically, we manually assigned probabilities to each action so that triggered surveys were not overly intrusive and there was an adequate sampling of infrequent actions (e.g., wall posts) compared to highly frequent ones (e.g., seeking in videos).

The survey was displayed in a pop-up window. Students had the option to ignore surveys. To decrease the prevalence of the surveys, once a survey was triggered for a student, the student was removed from the survey pool for two weeks. Our dataset includes surveys from the 2017-2018 school year (September through May). In this time, 69,174 students responded to at least one survey. The mean number of survey responses per student was 1.94 (median = 1). Of the students that responded, the minimum number of responses was 1 and the maximum number of responses by any student was 14.

Each survey targeted one affective state, randomly selected, from the following: Anxiety, Boredom, Confusion, Contentment, Curiosity, Disappointment, Engagement, Frustration, Happiness, Hopefulness, Interest, Pride, Relief, Sadness, Surprise, Mind Wandering, Pleasantness, and Wakefulness. We chose several states because they closely relate to learning [21] while others address core dimensions of affect such as valence and arousal [11]. Mind Wandering, Pleasantness, and Wakefulness represent bipolar concepts, so we used a seven-point scale with contrasting options and presented prompts for each polarity (e.g., sleepy/awake). The other states used a five-point scale ranging from “Not at all ___” to “Very ___”. In our analysis, we linearly scaled all survey responses to lie in a five-point range so that all states are represented equally.

2.3 Generic Activity Features

We recorded student activity on Algebra Nation using 22 features that did not depend on specific content (e.g. which video was watched or a particular quiz question). These activity features included attempting quizzes, watching videos, and interacting with the wall or discussion board. Based on our prior work [9], we counted the number of occurrences of each feature over 30-second chunks and summed the counts for each action across 5-minute window lengths preceding an affect survey. In some cases, the platform measured an unnaturally high amount of activity (e.g. playing/pausing a video 100 times within 30 seconds). We addressed these outliers by limiting each 30-second chunk to 10 recorded activities.

2.4 Usage Features

In addition to session-specific generic activity features, which were used to train the models, we were interested in investigating

generalizability to differences in how students interact with the platform over an entire school year. To do this, we defined five usage features. First, we calculated the proportion of sessions students use their mobile device compared to a desktop computer as this may indicate the context in which students are using the platform (e.g., at home or while commuting). Second, we calculated the proportion of sessions in the spring semester compared to the fall semester. We were interested in this feature because students must pass the algebra standardized exam that is offered in the spring semester in order to graduate from high school. To model how much students use the platform, we calculated the number of sessions and the average length of each session. Students may leave an active session open for long session times in their browser while switching to another task or repeatedly log into the platform without recording any meaningful interactions. We replaced these outliers with the 99th percentile value for the average length and number of logins. Finally, we calculated the mean time of day that students use the platform, which can indicate whether students primarily use the platform during the school day or at home. Usage data was available for 235,756 individual students, including those who did not receive or respond to any surveys.

2.5 Demographics

We also obtained records of demographic data of 118,177 students from the Florida Department of Education. This dataset includes students from grade 6 through grade 12, from which we defined three groups. We defined the first group as Middle School (54%), which includes grades 6 through 8. These students are often advanced and are enrolled in the Algebra course earlier than is typically expected by state standards [6]. The second group is Grade 9 (37%). We chose to keep this grade separate because it has one of the largest enrollment numbers and grade 9 is when students are enrolled in the course during the typical mathematics sequence. Lastly, we defined High School (9%) as grades 10 through 12. These students are often behind in the typical mathematics sequence and struggle to pass the course before they graduate. For gender, the available data classifies students as Male (49%) or Female (51%), which we took at face value.

This dataset records student eligibility for free or reduced-price (F/R) school lunch, which is one indicator of socioeconomic status (but see Harwell & LeBeau [8]). We defined the groups as F/R (53%) and Other (47%), with the latter reflecting those who did not qualify or did not apply. We combined free and reduced because there were so few students that qualified for a reduced-price lunch.

Finally, this dataset includes data on race and ethnicity. We defined these groups to approximately balance group size: White (72%), Black (23%), Hispanic (32%), and Other (13%; Asian, Native American, Pacific Islander, and Mixed).

3. CLUSTERING

We clustered participants based on usage characteristics and demographics to investigate the generalizability of the affect models across clusters. To determine the number of clusters, we inspected the dendrogram generated with Ward hierarchical clustering [31] using the SciPy library (<http://www.scipy.org/>). For efficient clustering, we randomly sampled 1,000 instances. We then used the k-means algorithm [14] to construct the clusters using scikit-learn [19]. We chose to use all available students regardless of their participation in the surveys since our goal was to generalize over as many students as possible.

We constructed usage clusters using the five features described in Section 2.4. We first scaled each of these features to [0, 1]. The above procedure yielded five clusters (Table 1). One group (U1) showed heavy usage patterns (signified by long sessions and numerous log-ins). Two groups were defined by primarily mobile sessions and were further differentiated by sessions focused in either the fall (U4) or spring (U5) semester. Finally, two groups showed particularly light usage patterns and were differentiated by sessions focused in either the fall (U3) or spring (U2) semester.

Next, we constructed clusters using the demographic features described in Section 2.5. We dummy encoded our variables resulting in seven features indicating grade level, three features indicating lunch status, seven features indicating race/ethnicity, and one feature indicating gender. The above procedures yielded seven clusters (Table 2). Grade level largely differentiated clusters. Only

Table 1. K-means cluster centers based on typical usage. Distinguishing features are bolded.

ID	Cluster Description	Session Time (min)	Num. Sessions	Prop. Spring Use	Prop. Desktop Use	Time of Day (hour)	Prop. of Users
U1	Spring semester, heavy use	45.46	25.44	0.75	0.90	14.19	0.20
U2	Spring semester, light use	14.26	3.53	0.96	0.99	14.86	0.35
U3	Fall semester, light use	13.81	3.46	0.11	0.99	14.81	0.28
U4	Fall semester, mobile use	21.38	9.54	0.19	0.30	13.25	0.07
U5	Spring semester, mobile use	29.66	10.95	0.93	0.27	13.21	0.10

Table 2. Demographic cluster centers. For clarity, only distinguishing features are displayed and are bolded.

ID	Cluster Description	Grade 7	Grade 8	Grade 9	Grade 10	F/R Lunch	White	Black	Asian	Prop. of Users
D1	Split grades, F/R lunch, Black	0.08	0.29	0.44	0.17	0.99	0.03	1.00	0.01	0.16
D2	Grade 7, not F/R lunch, White/Asian	1.00	0.00	0.00	0.00	0.20	0.80	0.06	0.16	0.10
D3	Grade 8, not F/R lunch, White	0.00	0.99	0.00	0.00	0.00	0.87	0.09	0.07	0.22
D4	Grade 8, F/R lunch, White	0.00	1.00	0.00	0.00	1.00	0.90	0.03	0.06	0.15
D5	Grade 9, F/R lunch, White	0.11	0.00	0.87	0.00	1.00	0.91	0.01	0.03	0.16
D6	Grade 9, not F/R lunch, White/Black	0.00	0.00	0.99	0.00	0.04	0.81	0.16	0.04	0.16
D7	Grade 10, split F/R lunch, White/Black	0.00	0.00	0.00	1.00	0.52	0.76	0.20	0.03	0.05

cluster D1 had a significant distribution of students across grade levels. Another differentiating feature was lunch status, where three clusters (D1, D4, D5) were largely comprised of students on F/R lunch. Four clusters were differentiated by race (D1, D2, D6, D7).

4. AFFECT DETECTION MODELS

4.1 Model-building Procedure

We used scikit-learn [19] to implement a supervised learning pipeline. We chose to use the Bayesian Ridge Regression algorithm [13] since it produced good overall results in previous work on the same data [9] compared to several more complicated alternatives.

We trained regression models using 10-fold student-level cross validation. For each fold, instances for each student were included in *either* the training or testing set. This practice reduces overfitting and increases the likelihood that the model will generalize to new students. In each fold, we trained a model using the generic activity features and generated predicted survey responses on the test data. We evaluated the performance of the model using the Spearman correlation as it assumes ordinal and continuous values. We then averaged these scores across folds to obtain a final accuracy score.

We trained prediction models for positive and negative affective valence rather than the original 18 states measured in the surveys. We initially trained a model for each state and calculated the correlation between the predicted survey responses for each state. These predictions were strongly correlated within positive and negative valence. We then trained positive and negative valence models using the combined set of states and generated predictions for the individual states. The mean performance of the valence models was similar to training individual affective models, so we chose to use the valence models for parsimony. For the positive valence models, we included the following states: Arousal, Contentment, Engagement, Happiness, Hopefulness, Interest, Pleasantness, Pride, and Relief. For the negative valence models, we included the following states: Anxiety, Boredom, Confusion, Disappointment, Frustration, Mind Wandering, and Sadness. We did not include Curiosity and Surprise since their valence does not clearly align on either direction.

4.2 Preliminary Models on Cluster Membership

We first investigated whether our models discriminated using group features rather than the generic activity features. To test this, we trained models using cluster membership as the training data instead of activity features. We expected these models to perform poorly since they are not simply reflecting group differences. Indeed, we found that the average Spearman correlations were low (between 0.02 and 0.05) for both cluster models.

4.3 Generalizability

Our main analysis focused on investigating how our models, trained on activity features, generalize across different clusters. First, we considered a general model trained on the entire dataset using 10-fold student-level cross validation. We then built cluster-specific models. For each cluster, we trained and tested a model on data from that cluster. We also tested this model on the other cluster data. For example, we trained a model on U1 and tested on each of the other clusters (U2 – U5) as well as the entire dataset (All). We performed this procedure separately for the positive and negative valence states as well as for the usage and demographic clusters¹.

¹ Similar results for other slices can be found using this code (link).

4.4 Results

We examined the generalizability of our models using the procedure in Section 4.3. If our models generalized well, we expect to see a model trained on one group perform similarly well when applied to other groups (Table 3). This was the case for the usage clusters, where the maximum difference between testing on one cluster and testing on another is 0.05. The demographic clusters were more varied. In this case, the greatest difference between testing on the one cluster and testing on another was 0.09.

Recent metrics proposed in slicing analysis, such as [7], apply to classification problems and not the regression task considered here. To better quantify model generalizability, we defined an *individual advantage* metric. Using the procedure from Section 4.1, we trained a model using the training set X and tested the model using the testing set Y . We represented the performance of the model, which is the average Spearman correlation, as $P_{X,Y}$. For a target group T , we defined the individual advantage metric as $(P_{T,T} - P_{All,T})/P_{All,T}$. This describes the proportion improvement over using a general model for the target group T . Therefore, a perfectly generalizable model would have an individual advantage of 0 since an individual model and general group model will have the same accuracy.

We used this metric to quantify the generalizability of our models. Both positive and negative models showed small, positive individual advantage values (mean usage 0.04; mean demographics 0.02), which indicates a small advantage to training cluster-specific models compared to a general model.

5. SAMPLE SIZE SIMULATIONS

We then investigated whether sample size affects the advantage for using individual models. Specifically, are individual advantages mitigated when more data is available? To address this question, we computed the average individual advantage metric over 10 cross validation folds for a range of sample sizes starting at 500. For each sample size, we randomly selected the appropriate number of instances from the training sets. We incrementally increased the sample size by 200 until we reached the actual group size, which varied between 1,500 and 7,100. We repeated this simulation 1,000 times and calculated the 95% confidence interval of the mean individual advantage metric at each sample size (Figure 1).

Table 3. Mean correlation of positive valence models (negative in parentheses) for usage clusters.

	Test U1	Test U2	Test U3	Test U4	Test U5	Test All
Train U1	0.25 (0.19)	0.23 (0.22)	0.21 (0.19)	0.19 (0.19)	0.21 (0.18)	0.21 (0.20)
Train U2	0.21 (0.18)	0.26 (0.23)	0.22 (0.19)	0.21 (0.20)	0.22 (0.19)	0.22 (0.20)
Train U3	0.21 (0.17)	0.24 (0.21)	0.23 (0.20)	0.20 (0.19)	0.20 (0.17)	0.22 (0.19)
Train U4	0.20 (0.17)	0.25 (0.22)	0.21 (0.19)	0.21 (0.20)	0.21 (0.19)	0.22 (0.21)
Train U5	0.23 (0.17)	0.25 (0.22)	0.21 (0.18)	0.20 (0.19)	0.24 (0.21)	0.22 (0.20)
Train All	0.22 (0.18)	0.25 (0.22)	0.22 (0.20)	0.21 (0.20)	0.22 (0.19)	0.22 (0.21)

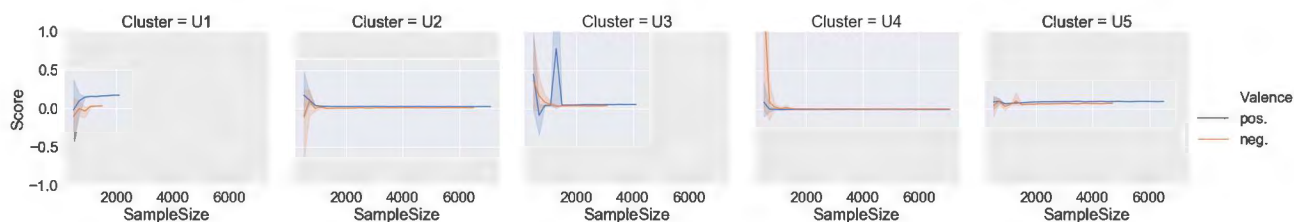


Figure 1. Averaged individual advantage simulation scores for usage clusters

We first noted that the scores for models using sample sizes less than 1,500 varied wildly, as indicated by the width of the confidence intervals in this region. As such, we can only conclude that results obtained from small samples might not be reliable. This is concerning since previous work used sample sizes ranging from 20 to 646 students [9]. As expected, for larger sample sizes, the models quickly stabilized and produced more reliable scores.

For most clusters, the individual advantage scores stabilized to a value of 0.10 or less, which indicates a small advantage of training cluster-specific models. The scores for clusters D1 and U1 seemed to increase as the sample size increased, but we cannot make strong conclusions since the sample size of these clusters was small.

6. DISCUSSION

In an attempt to answer the call for predictive generalizability [7, 10, 26], we used interaction data from 69,174 students over an entire school year to study the extent to which sensor-free affect detectors generalize across usage and demographic clusters.

6.1 Main Findings

We found that students primarily differed in their interaction rate, active semester, and primary device. The demographic clusters were primarily discriminated by grade level, F/R lunch eligibility, and (to a smaller extent) race. Using cluster membership as the only training feature resulted in near-zero results, which shows students in a particular cluster are not generally predisposed to certain affective states. We must then consider the context of a student's activity when predicting their immediate affective state.

Similar to previous work [1, 4, 9, 17, 23, 24, 27, 28], we examined the generalizability of our models by training cluster-specific models and testing them on the other clusters. We found that cluster-specific models perform slightly better on the target cluster, with a maximum difference of 0.05 for the usage clusters and 0.09 for the demographic clusters. We expanded this analysis by introducing an individual advantage metric, which measures the advantage given to a target group compared to a general (entire population) model. This metric agreed with our initial analysis by showing a small advantage given by training a cluster-specific model. The maximum advantage was 0.14 for the usage clusters and 0.11 for the demographic clusters. Although these results provide evidence that cluster-specific models are better at predicting affective valence, it is not clear what difference is meaningful in practice.

Lastly, we investigated how model generalizability changes in response to sample size. We performed a series of simulations that trained affect-detection models and systematically varied sample sizes. Models trained on 1,500 samples or less did not generate stable scores or predictions, even after 1,000 iterations. When considering sample sizes greater than 1,500 that yielded reliable scores, we found that the individual advantage scores stabilize as sample size increases at a value of 0.10 or less, which is consistent with our initial analysis. This suggests that generalizability is not greatly affected by sample size beyond the 1,500-sample threshold.

6.2 Limitations and Future Work

The greatest area of improvement is the overall model performance. As discussed in [9], the average performance corresponds to a small-sized effect. This is likely caused by the limited number and extreme generality of the training features. Future work can address this by introducing more platform-specific features, such as which quiz a student was attempting. We can then see if our models have the power to distinguish between individual affective states rather than simply identifying positive or negative valence. Of course, the use of these features will result in more platform-specific models, which limits their generalizability to different platforms or even to other domains within the sample platform.

Our analysis of generalizability was limited to demographic features and overall interaction with the Algebra Nation platform. This analysis should be extended to include other academic subjects, time frames, and regional groups. For example, while Florida does reflect the overall demographic composition of the United States, other states do not. It would be interesting to see how our models generalize to other populations. With respect to subject generalizability, while [9] showed generalizability between Algebra and Geometry, we could see how a model for mathematics generalizes to unrelated subjects such as chemistry or music.

There are several exciting opportunities to apply these large-scale sensor-free affect detectors. First, we will be able to develop real-time interventions based on predictions of a student's affective state and promote more a more engaging experience with the curriculum. In addition, as we collect data from different regions and over longer time periods, we can more directly investigate the relationship between engagement and end-of-course scores.

Lastly, it is important to understand the impacts of a one-size-fits-all model on long-term student achievement. When developing interventions, one should consider possible effects if predictions of affect are incorrect. In this case, the intervention should not have any negative consequences for the student receiving it.

7. CONCLUSION

Sensor-free affect detection models provide the opportunity to provide personalized experience for large populations of students. In this work, we answered the call to investigate how these predictive models generalize between different subpopulations in the training data. We did this using a longitudinal dataset of student interaction with an online math learning platform with our groups of interest being clusters based on typical usage on the platform and demographic features. We showed that while models trained on one cluster perform similarly well when applied to the other clusters, there is a small advantage to use individual subpopulation models rather than one general population model. It is important to consider these models' differential performance and impact when deploying large-scale platforms that adapt to sensor-free predictions of individual students' affective states.

8. ACKNOWLEDGMENTS

This work was supported by the Institute of Education Sciences, U.S. Department of Education, through grant R305C160004 and Intel Research. The opinions expressed are those of the authors and do not represent views of the Institute, the U.S. Department of Education, or Intel.

REFERENCES

- [1] Baker, R.S.J. d. et al. 2008. Developing a generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction*. 18, 3 (Aug. 2008), 287–314.
- [2] Baker, R.S.J. d. and Gowda, S.M. 2010. An analysis of the differences in the frequency of students’ disengagement in urban, rural, and suburban high schools. *Proceedings of the 3rd International Conference on Educational Data Mining* (Pittsburgh, Pennsylvania, 2010), 11–20.
- [3] Baker, R.S.J. d. and Ocumpaugh, J. 2014. Interaction-based affect detection in educational software. *The Oxford Handbook of Affective Computing*. R.A. Calvo et al., eds. Oxford University Press.
- [4] Bosch, N. et al. 2015. Temporal generalizability of face-based affect detection in noisy classroom environments. *Artificial Intelligence in Education* (2015), 44–53.
- [5] D’Mello, S. et al. 2014. I feel your pain: a selective review of affect-sensitive instructional strategies. *Design Recommendations for Intelligent Tutoring Systems - Volume 2 Instructional Management*. R.A. Sottilare et al., eds. U.S. Army Research Laboratory. 35–48.
- [6] Education, F.D. of 2014. Mathematics florida standards (mafs).
- [7] Gardner, J. et al. 2019. Evaluating the fairness of predictive student models through slicing analysis. *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (2019), 225–234..
- [8] Harwell, M. and Lebeau, B. 2010. Student eligibility for a free lunch as an ses measure in education research. *Educational Researcher*. 39, 2 (2010), 120–131.
- [9] Hutt, S. et al. 2019. Time to scale : generalizable affect detection for tens of thousands of students across an entire school year. *2019 CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019)* (2019).
- [10] Kusner, M. et al. 2017. Counterfactual fairness. *31st Conference on Neural Information Processing Systems (NIPS 2017)* (2017), 4066–4076.
- [11] Linnenbrink, E.A. 2007. The role of affect in student learning: a multi-dimensional approach to considering the interaction of affect, motivation, and engagement. *Emotion in Education*. P.A. Schutz and R. Pekrun, eds. Elsevier Inc. 107–124.
- [12] Liyanagunawardena, T.R. et al. 2013. MOOCs: a systematic study of the published literature 2008-2012. *International Review of Research in Open and Distance Learning*. (2013).
- [13] MacKay, D.J.C. 1992. Bayesian interpolation. *Neural Computation*. 4, 3 (May 1992), 415–447.
- [14] MacQueen, J. 1967. Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (Berkeley, California, 1967), 281–297.
- [15] Moos, D.C. 2014. Setting the stage for the metacognition during hypermedia learning: what motivation constructs matter? *Computers and Education*. 70, (2014), 128–137.
- [16] Nicaud, J.F. et al. 2006. Experiments with aplusix in four countries. *International Journal for Technology in Mathematics Education*. 13, 2 (2006), 79–88.
- [17] Ocumpaugh, J. et al. 2014. Population validity for educational data mining models: a case study in affect detection. *British Journal of Educational Technology*. 45, 3 (May 2014), 487–501.
- [18] Ogan, A. et al. 2012. Collaboration in cognitive tutor use in latin america: field study and design recommendations. *2012 CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2012)* (New York, New York, USA, 2012), 1381–1390.
- [19] Pedregosa, F. et al. 2011. Scikit-learn: machine learning in python. *Journal of Machine Learning Research*. 12, (2011), 2825–2830.
- [20] Pekrun, R. 2017. Emotion and achievement during adolescence. *Child Development Perspectives*. 11, 3 (2017), 215–221.
- [21] Pekrun, R. 2007. Emotions in students’ scholastic development. *The Scholarship of Teaching and Learning in Higher Education: An Evidence-Based Perspective*. R.P. Perry and J.C. Smart, eds. Springer. 553–610.
- [22] Porayska-Pomsta, K. et al. 2013. Knowledge elicitation methods for affect modelling in education. *International Journal of Artificial Intelligence in Education*. 22, 3 (2013), 107–140.
- [23] Samei, B. et al. 2015. Modeling classroom discourse: do models that predict dialogic instruction properties generalize across populations? *Proceedings of the 8th International Conference on Educational Data Mining* (2015), 444–447.
- [24] San Pedro, M.O.C.Z. et al. 2011. Detecting carelessness through contextual estimation of slip probabilities among students using an intelligent tutor for mathematics. *Artificial Intelligence in Education* (2011), 304–311.
- [25] Sandeen, C. 2013. Integrating moocs into traditional higher education: the emerging “mooc 3.0” era. *Change: The Magazine of Higher Learning*. 45, 6 (2013), 34–39.
- [26] Sculley, D. et al. 2018. Winner’s curse? on pace, progress, and empirical rigor. *6th International Conference on Learning Representations* (2018).
- [27] Soriano, J.C.A. et al. 2012. A cross-cultural comparison of effective help-seeking behavior among students using an its for math. *Intelligent Tutoring Systems* (2012), 636–637.
- [28] Stewart, A. et al. 2017. Generalizability of face-based mind wandering detection across task contexts. *Proceedings of the 10th International Conference on Educational Data Mining* (2017), 88–95.
- [29] U.S. Census Bureau 2018. *Florida QuickFacts*.
- [30] U.S. Census Bureau 2018. *United States QuickFacts*.
- [31] Ward, J.H. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*. 58, 301 (Mar. 1963), 236–244.

Time-series Insights into the Process of Passing or Failing Online University Courses using Neural-Induced Interpretable Student States

Byungsoo Jeon
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA
byungsoj@cs.cmu.edu

Eyal Shafran
Western Governors University
4001 S 700 East
Salt Lake City, UT
eyal.shafran@wgu.edu

Luke Breitfeller
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA
mbreitfe@cs.cmu.edu

Jason Levin
Western Governors University
4001 S 700 East
Salt Lake City, UT
jason.levin@wgu.edu

Carolyn P. Rosé
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA
cprose@cs.cmu.edu

ABSTRACT

This paper addresses a key challenge in Educational Data Mining, namely to model student behavioral trajectories in order to provide a means for identifying students most at-risk, with the goal of providing supportive interventions. While many forms of data including clickstream data or data from sensors have been used extensively in time series models for such purposes, in this paper we explore the use of textual data, which is sometimes available in the records of students at large, online universities. We propose a time series model that constructs an evolving student state representation using both clickstream data and a signal extracted from the textual notes recorded by human mentors assigned to each student. We explore how the addition of this textual data improves both the predictive power of student states for the purpose of identifying students at risk for course failure as well as for providing interpretable insights about student course engagement processes.

Keywords

Student State, Clickstream Data, Mentor's Notes, LDA, Time-series Modeling, Deep Learning

1. INTRODUCTION

In online universities, modeling the population of students at scale is an important challenge, for example, in order to identify students most at-risk and to provide appropriate interventions to improve their chances of earning a degree in a timely fashion. In this respect, a plethora of approaches for clickstream analysis [11, 21, 22] have been published in

the field of Educational Data Mining, which address questions about modeling student course engagement processes [19]. While clickstream data is the most readily available, and while some success has been achieved using it for this purpose, its low level indicators provide only glimpses related to student progress, challenges, and affect as we would hope to observe and model them. In this paper, we explore the extent to which we may achieve richer insights by adding textual data to the foundation provided by clickstream data.

One advantage to modeling student behavior and states from a for-pay platform is that the level of support provided to students is greater than in freely available contexts like Massive Open Online Courses (MOOCs), and this more intensive engagement provides richer data sources that can be leveraged. In our work, we make use of a new data source provided by the Western Governor's University (WGU) platform, where each student is assigned a human mentor, and the notes from each biweekly encounter between student and mentor are recorded and made part of the time series data available for each student. Thus, even if we do not have access to the full transcript of the interactions between students and their mentors, we can leverage the documentation of provided support in order to enhance the richness and ultimately the interpretability of student states we may induce from other low level behavioral indicators we can extract from traces of learning platform interactions.

A major thrust of our work has been to develop a technique for leveraging this form of available textual data. We refer to this data as *Mentor's Notes*. In particular, we propose a sequence model to integrate available data traces over time, **Click2State**, which serves a dual purpose. The first aim is to induce predictive student states, which provide substantial traction towards predicting whether a student is on a path towards passing or failing a course. Another is to provide us with insights into the process of passing or failing a course over time, and in particular leveraging the insights of human mentors whose observations give deeper meaning to the click level behavioral data, which is otherwise impoverished from an interpretability standpoint.

Byungsoo Jeon, Eyal Shafran, Luke Breitfeller, Jason Levin and Carolyn P. Rose "Time-series Insights into the Process of Passing or Failing Online University Courses using Neural-Induced Interpretable Student States" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 330 - 335

In the remainder of the paper we motivate our specific work as situated within the literature. Next we present our modeling approach and a series of experiments that investigate the following three research questions: (RQ1) How can we extract information and meaning from mentors' notes about the formation of student states across time? (RQ2) To what extent does integrating a representation of topical insights from *Mentor's Notes* improve the ability of a time series neural model to predict whether students are on a path towards passing or failing a course? (RQ3) How can we use insights about student progress in an online course captured using student state representations from our model to understand the process of passing or failing a course? The more comprehensive version of this paper is available at Arxiv ¹.

2. RELATED WORK

One of the most important challenges in providing analytic tools for teachers and administrators [1, 19] in online university is to model the population of students in such a way as to provide both predictive power for triggering interventions and interpretability for ensuring validity. Some past research has already produced models to identify at-risk students and predict student outcomes specifically in online universities [6, 16]. For example, Smith et al. [20] proposed models to predict students' course outcomes and to identify factors that led to student success in online university courses. Eagle et al. [10] presented exploratory models to predict outcomes like overall probability of passing a course, and provided examples of strong indicators of student success in the WGU platform where our work is also situated. However, this past work has focused mainly on predictive modeling of student outcomes, whereas our work pursues both predictive power and interpretability.

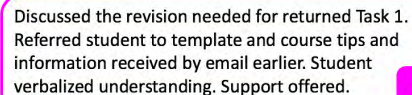
While much work in the field of Educational Data Mining explores time series modeling and induction of student state representations from open online platforms such as Massive Open Online Courses (MOOCs) or Intelligent Tutoring Systems, far less has been published from large, online universities such as WGU, which offer complementary insights to the field. Student states are triggered by students' interaction with university resources, their progress through course milestones, test outcomes, affect-inducing experiences, and so on. Affect signals in particular have been utilized by many researchers as the basis for induced student states, as this rich source of insight into student experiences has been proven to correlate with several indicators of student accomplishments [18]. Researchers have investigated affect and developed corresponding detectors using sensors, field observation, and self-reported affect. These detectors capture students' affective signals from vocal patterns [7, 17], posture [9], facial expressions [3, 17], interaction with the platform [4, 5, 12], and physiological cues [7, 15]. Although these signals provide rich insights, the requisite data is sometimes expensive or even impractical to obtain, even on for-pay platforms such as WGU, where we conduct our research.

The bulk of existing work using sequence modeling to induce student states has focused on the data that is most readily available, specifically, clickstream data. For example, Tang et al. [21] have constructed a model to predict a set of stu-

dent actions with long short-term memory (LSTM) [13] on student clickstream data from a BerkeleyX MOOC, though the basic LSTM was unable to match the baseline of defaulting to the majority class for samples of student actions. Fei et al. [11] proposed a sequence model to predict dropout based on clickstream data using recurrent neural network (RNNs) model, with more success. Wang et al. [22] also built a neural architecture using a mix of convolutional neural network (CNN) [14] and RNN for dropout prediction from clickstream data. Though these models have achieved differing success at their predictive tasks, a shared shortcoming is the lack of interpretability in the induced student state representations. Our work extends previous studies by proposing a model that enriches temporal signals from clickstream data using the textual mentor's notes to provide a means for interpreting student state representations.

3. DATA

Our study is based on data collected by Western Governor's University (WGU), an online educational platform ². To support self-paced learning, students in WGU are assigned to a program mentor (PM). The PM is in charge of evaluating a student's progress through their degree and helping to manage obstacles the student faces. A PM and a student generally have bi-weekly live calls, but this may vary depending on the student's needs and schedule. Each PM writes down a summary of what was discussed, which we refer to as a mentor's note. An example is given in Figure 1. Mentor's notes describe the status and progress of the student and what types of support was offered or what suggestions were made during the call. This information can provide meaningful cues to infer student states over time.



Discussed the revision needed for returned Task 1. Referred student to template and course tips and information received by email earlier. Student verbalized understanding. Support offered.

Figure 1: An example of mentor's notes.

In this work we specifically investigate how the use of the mentor's note data alongside the more frequently used clickstream data might enable that important goal. Clickstream data in WGU also provides us with information on how active students are and where in the WGU platform they spend their time. We collect clickstream data from four different types of web pages in the WGU platform: course, degree plan, homepage, and portal. The course web pages cover all pages related to courses in WGU. Degree plan represents a dashboard where students check their progress toward a degree. Homepage is the main page that shows students' progress in each course and allows access to all provided WGU features. Portal covers any other pages for student support including technical and financial assistance.

An example of the clickstream data can be seen in Table 1. Each row represents one of five different click sources: target course page, other course page, degree plan page, portal page, and homepage. We divide the course pages into "target course" and "other course". Each column represents one of different six click types: click count (C1), focus state count (C2), keypress count (C3), mousemove count (C4), scroll

¹<http://arxiv.org/abs/1905.00422>

²<https://www.wgu.edu/>

count (C5), and unfocused state count (C6). The values in the table represent the weekly count of different type of clicks from each different source.

For this paper, we have collected the mentor’s notes and clickstream data from two courses conducted in 2017: Health Assessment (HA) and College Algebra (CA). We choose these two courses because they are popular among students and represent different levels of overall difficulty. Table 2 shows the statistics for the dataset. “Average prior units” is the average number of units students transferred to WGU from prior education when they started the degree, and functions as a proxy for the level of student’s prior knowledge. We split the dataset for each course into a training set (80%), a validation set (10%), and a test set (10%). For training, to avoid a tendency for trained models to over-predict the majority class, we have resampled the training set so that both the pass state and the fail state are represented equally.

	C1	C2	C3	C4	C5	C6
Target course	53	61	0	168	904	1732
Other courses	177	167	0	455	2301	4887
Degree plan	0	0	0	0	0	0
Portal	21	89	0	263	3862	2440
Homepage	36	69	0	122	72	1581

Table 1: Example of clickstream data.

	HA	CA
# of students	6,041	4,062
Length of a term	25 weeks	25 weeks
Avg prior units	62 ± 39	11 ± 23
Fail rate	0.185	0.509
Avg # of notes per student	10.9 ± 5.7	11.0 ± 5.8
Avg length of notes (chars)	198 ± 47	194 ± 55

Table 2: Data Statistics

4. PROPOSED METHOD

As we have stated above, in our modeling work, we propose a sequence model, **Click2State**, with two primary purposes. The first is to form a student state representation that will allow us to better identify students at risk of failing a course than a baseline model that does not make use of rich textual data. The second is to provide us with a means to interpret the meaning of a student state representation.

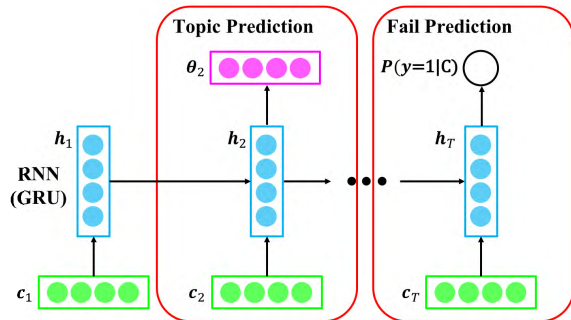


Figure 2: Architecture of Click2State Model.

Figure 2 provides a schematic overview of our proposed model. Note that it is first and foremost a sequence model that predicts whether a student will pass or fail a course

based on an interpretable student state that evolves from week to week as each week’s clickstream data is input to the recurrent neural model. A summary of the content of the mentor’s note for a week is constructed using a popular topic modeling technique, specifically Latent Dirichlet Allocation (LDA) [2]. In the full model, an intermittent task to predict the topic distribution extracted from the mentor’s notes associated with a time point is introduced. The goal is to use this secondary task to both improve the predictive power of the induced student states over the baseline as well as to enhance the interpretability of the state representation.

Feature Vector Design We train our model using clickstream feature vectors (as input) and topic distribution vectors (as output for the topic prediction task). We design the clickstream feature vector to include both an encoding of click behavior of students from a time period as well as a control variable that represents the prior knowledge of students as estimated by the number of units they were able to transfer in. The full clickstream feature vector contains thirty weekly counts for each different type and source of click, in addition to the single control variable just mentioned, which is the number of transferred units. We use min-max normalization to scale the values between 0 and 1. To extract a topic distribution vector for each mentor’s note, we run Latent Dirichlet Allocation (LDA) over the whole set of mentor’s notes from the entire training dataset.

Formal Definition of the Model Denote the student’s clickstream features by $C = (c_1, c_2, \dots, c_T)$, where c_t is the clickstream feature vector of t th week, and T is the number of weeks for the term. The clickstream feature vectors are encoded via Gated Recurrent Units (GRU) [8], which are variants of the Recurrent Neural Network (RNN). At each time step t , this network constructs a hidden state of the student for the t th week, $h_t \in \mathbb{R}^H$, where H is the dimensionality of the hidden state. We consider h_t as the student state representation at t th week. Based on the generated student state representation from RNN (h_t), our model is trained to predict a topic distribution of a mentor’s note and the probability of the student failing that course.

Topic Prediction Given the generated hidden states from RNN (h_t) for the t th week, the model estimates the true topic distribution ($\theta_t \in \mathbb{R}^{N_t}$) of a mentor’s note on t th week where N_t is the number of topics. The estimated topic distribution ($\hat{\theta}_t \in \mathbb{R}^{N_t}$) is computed by taking h_t as an input of one fully connected layer (weight matrix: W_θ) whose output dimensionality is N_t followed by a softmax layer.

$$\hat{\theta}_t = \text{Softmax}(W_\theta h_t)$$

Fail Prediction As data from a student’s participation in a course is fed into the RNN week by week, the model estimates the probability of the student failing that course ($P(y = 1|C)$) at the last timestep T . The estimated probability is computed by taking h_T as an input of one fully connected layer (weight matrix: W_y) whose output dimensionality is one followed by a sigmoid layer.

$$P(y = 1|C) = \text{Sigmoid}(W_y h_T)$$

Loss The loss function is composed of KL divergence loss for the topic prediction and binary cross-entropy loss for the fail prediction. Assume there are a total of N students. The

KL divergence loss of topic distribution of the mentor's note for n th student at time t is defined as:

$$KLD_{n,t} = D_{KL}(\theta_{n,t} \parallel \hat{\theta}_{n,t}),$$

where $\theta_{n,t}$ and $\hat{\theta}_{n,t}$ are the true and estimated topic distribution of the mentor's note at time t for n th student. The binary cross-entropy of the n th student measures the similarity between $P(y_n = 1|C)$ and the true y_n as:

$$BCE_n = -y_n \log P(y_n = 1|C) - (1 - y_n) \log(1 - P(y_n = 1|C)),$$

Assume that there are a total of N_n mentor's notes for n th student. Combining the two losses, our final loss is

$$\frac{1}{N} \sum_{n=1}^N [\lambda BCE_n + (1 - \lambda) \frac{1}{N_n} \sum_{t=t_{n,1}}^{t_{n,N_n}} KLD_{n,t}],$$

where $t_{n,i}$ is the timestep when n th student has i th mentor's note, and λ is the rescaling weight.

5. RESULTS

In this section, we answer our aforementioned research questions one by one with the experiment results.

RQ1. What types of information about student states can we extract from mentor's notes? We answer the question of how student state information may be extracted from mentor's notes through application of LDA to the notes. We set the number of topics to ten to maximize the interpretability of the results. Table 3 shows the learned topics with manually assigned labels, topical words, and text. Topical words are the top ten words with the highest probability of appearing within each learned topic, and are presented in decreasing order of likelihood. The topical text column contains an example snippet from one of top ten mentor's notes for each topic. We exclude the one topic that was incoherent out of the 10 learned topics.

Note that there are four topics related to student progress and plan, term plan (T5), course progress (T6), term progress (T7), and goal setting (T9). Course progress and goal setting (T6, T9) focus on progress towards modules in a particular course, along with past and present goals about the course itself. Term plan and term progress (T5, T7) emphasize discussions about plans for a term, such as courseload within in a term, course selection, and long-term degree planning. There is a clear utility to these topics as an interpretation tool for regulation of the student's process moving through the curriculum—if a student hits an impasse, mentor's notes are expected to focus on what challenges the student experienced and how to address these challenges.

The remaining six topics provide insight on specific issues and circumstances a student may be facing at a particular time, and which may end up impacting their overall progress. In revision (T1), we discover students seeking feedback on revisions, suggesting significant engagement with the platform. In question (T2), students ask for tips on using WGU platforms, course logistics, and how to succeed in a given course. In time constraint (T8), students point out time constraints in their daily life to explain why goals were not met. The time constraint (T8) topic may explain

abnormal absence or dropout. Assessment (T3) contains the result or plan of assessments and review for exam (T4) includes progress or plans of review for exam preparation.

RQ2. Does the task of topic prediction construct better student state representation than our baseline, as evaluated by the ability to predict student failure? We measure the predictive power of learned student state representation from our model and compare with that of our baseline, which shares the same neural architecture but is not trained on the extra task of topic prediction. The specific predictive task is to determine whether a student fails a course within a given term given a sequence of weeks of student clickstream data. We trained separate models to make a prediction after a set number of weeks so that we could evaluate the difference in predictive performance depending on how many weeks worth of data were used in the prediction. We measure the AUC scores of our model and baseline using data from two WGU courses: Health Assessment (HA) and College Algebra (CA).

Figure 3(a) shows the AUC scores across time steps for the HA course while Figure 3(b) shows the AUC scores across time steps for the CA course. For HA, our model achieves a statistically significant improvement (p-value < 0.05) in performance over the baseline model after the 5th week. For CA, our model achieves a statistically significant improvement after the 17th week. This difference in model performance between the HA and CA courses suggests the result from CA-specific topic data adds limited predictive power to the model. It is possible the clickstream data of students taking CA already contains enough information about whether a student is going to fail, a conclusion supported by the fact that AUC scores of the baseline model for CA are always better across time steps than those for HA.

Figure 3(c) exhibits the minimum KL Divergence loss of our model across time steps to determine how well our model is predicting the topic distribution of each mentor's note for each course. Though we determined that adding this task improves the fail prediction task, results on this task specifically are not impressive, demonstrating the relative difficulty of predicting mentor's notes from click data.

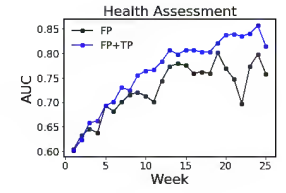
RQ3. What insights do we gain about the process of passing or failing a course over time from predicted mentor's notes topic distributions over time from the model? We perform two different experiments on the dataset of clickstream and mentors' notes data of students taking the College Algebra course. We choose this course because our topic prediction loss was lower (and thus, accuracy higher) for the course. First, we determine what topics inferred from our model correlate with whether a student will pass or fail a course. Then we find sequences of standardized topic probabilities of each topic inferred by our model that characterize students likely to pass or fail.

State	Assessment	Course progress	Term progress
P	-0.1486	0.6301	0.2298
F	0.1735	-0.0049	-0.1647

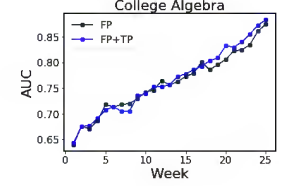
Table 4: Standard Score of Inferred Topic Probabilities from P and F State

Topic	Topical Words	Topical Text
T1. Revision	task, submit, revise, discuss, equate, complete, need, write, practice, paper	The ST and I discussed his Task 3 revisions after he made some corrections. The ST still needs to revise the task based on the evaluator's comments. He plans to do more revisions that align with the task rubric and submit the task soon.
T2. Question	student, question, call, email, send, course, discuss, appoint, speak, assist	Student emailed for help with getting started. CM called to offer support. Student could not talk for long. CM emailed welcome letter and scheduling link and encouraged for student to make an appointment
T3. Assessment	week, goal, today, schedule, pass, take, exam, final, work, talk	C278: took and did not pass preassessment, did not take final . NNP C713: took and did not pass the preassessment. Passed LMC1 PA with a 65 on 02/27. LMC1 exam scheduled for 02/27
T4. Review for exam	student, review, assess, plan, study, attempt, discuss, complete, take, report	Student scheduled appointment to review for first OA attempt but had taken and not passed the attempt by the time of the appointment.
T5. Term plan	student, discuss, course, complete, engage, college, term, plan, pass, progress	Discussed final term courses . Discussed starting C229 and working through hours and then working through C349 course .
T6. Course progress	goal, course, progress, current, complete, previous, work, date, pass, module	Current course : C349 Previous goal: completed modules 1-3 and engage shadow health by next appt date Progress toward goal: Yes New Goal: shadow health completed and engaged in video assessment
T7. Term progress	term, course, complete, date, goal, week, progress, current, leave, remain	Date: 8/22/17 Term Ends: 5 weeks OTP Progress : 5/14 cu completed Engaged Course : C785 Goal Progress : did not pass PA
T8. Time constraint	work, week, lesson, complete, go, progress, plan, finish, time, goal	NNP stated he was not able to make forward progress in course related to personal situation and time constraints from an unexpected event.
T9. Goal setting	goal, week, work, complete, task, progress, pass, accomplish, finish, contact	Previous goal : finish shadow health, finish and submit video by next call, start c228 next Progress /concerns: states working on c349 SH, discussed deadlines Goal : finish shadow health

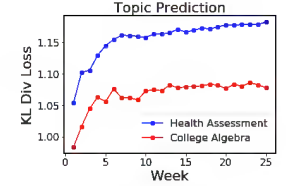
Table 3: LDA Topics Learned From Mentor's Notes



(a) Fail Prediction Performance on HA



(b) Fail Prediction Performance on CA



(c) Topic Prediction Performance

Figure 3: Performance of Fail and Topic Prediction

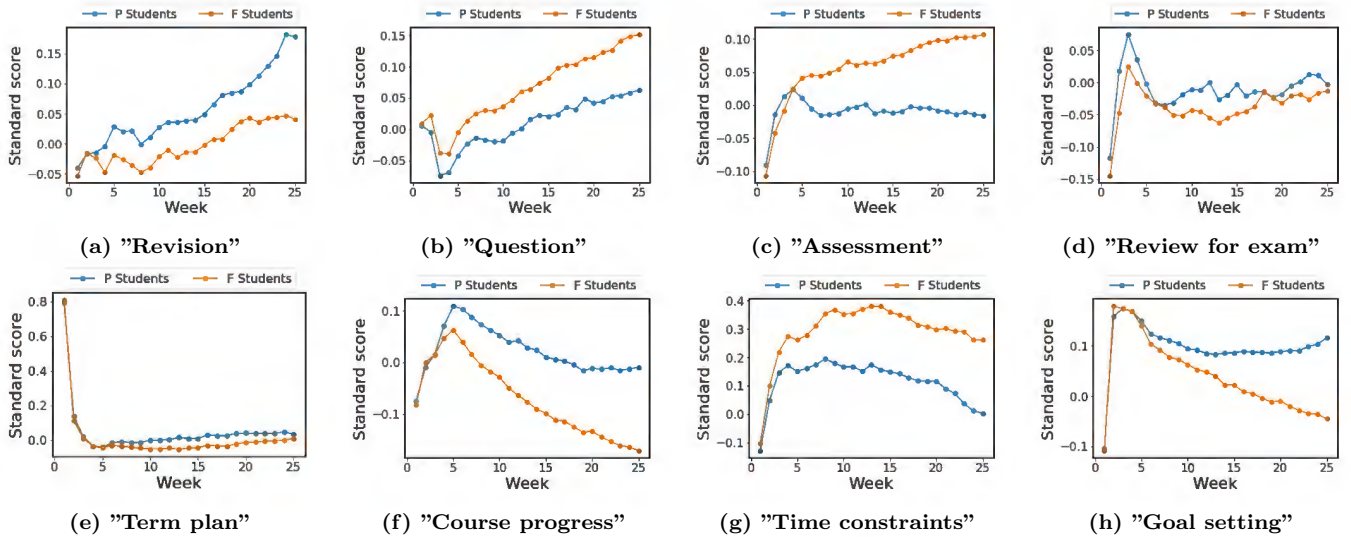


Figure 4: Standard Score of Each Topic Probability across Weeks for P and F Students

Experiment 1. In the first experiment, we find the two student state representations which minimize or maximize the probability of failing a course. We call the state representations that minimize and maximize the probability of failure as a P state and F state. Then, we show what topic distributions are inferred from each state. We represent emphasis, or a lack thereof, on a topic by standardizing topic probabilities and observing the number of standard deviations above and below the mean of a topic probability (standard score).

Table 4 shows the standard score for inferred topic probability from the P and F state. We only present the standard score of topics that vary wildly between P and F state. For example, the standard score of assessment topic (T3) for the P state is negative and for the F state is positive. One interpretation is that students likely to fail have more trouble in passing assessments, and thus talked to their mentors more about assessment topic (T3). The standard score of course and term progress (T6, T7) for the P state is positive and

negative for the F state, which shows students likely to pass report smoother progress instead of ongoing issues.

Experiment 2. We compare the trajectory of inferred probability of each topic by our model from students who passed (P students) and failed (F students) a course. Figure 4 shows the average standard score of topic probability per topic for P and F students over time.

We can see through this experiment clear, distinct patterns for the frequency of each topic over time that make intuitive sense given the format of online courses. For example, term plan (T5) is high frequency for the first week and plunges right after, since most students and mentors will naturally discuss plans for a term at the start of each term. The standard scores of other topics related to goal and progress (T6,T7,T9) also decrease over time, likely for similar reasons; the plot for T7 is omitted to save space, but it shows the similar pattern as T6. The standard scores of revision (T1), question (T2), and assessment (T3), meanwhile, increase over time, which may indicate students seek help more actively as they approach the end of a term. The standard scores of review for exam (T4) increase dramatically until the third week, decrease for few weeks, and finally level off. As the only condition for students in WGU to pass a course is to pass the final assessment, it may be that many students take their final assessments during the earlier weeks so they can pass a course as early as possible. The standard scores of time constraints (T8) steeply increase until the fourth week, and then gradually decrease over time. This suggests that when students begin a term they do not expect to have time constraints, but accumulate unanticipated issues in their personal lives as the course goes on.

For most topics, the P and F students exhibit distinct divergences in topic patterns. For topics related to goal and progress (T6, T7, T9), the gap between P and F students increases over time—suggesting that as time goes on F students will be reporting obstacles to their mentors instead of positive progress. The gap between P and F students for question (T2) increases over time, likely for similar reasons. For revision (T1), P students generally have higher standard scores than F students over time, supporting the idea that P students actively seek opportunities for revision towards the end of a term. For assessment (T3), standard score for F students increases over time while score for P students decreases. This could suggest that F students are more likely to procrastinate and struggle with their assessments than P students. Finally, for time constraints (T8) F students show higher standard score as time goes on. A likely interpretation is that students who encounter time constraints cannot devote focus to a course and are more likely to fail.

6. CONCLUSION

In this paper, we propose and evaluate a sequence model, **Click2State**, which aims to build an interpretable student state representation by leveraging mentor’s notes to give deeper meaning to impoverished clickstream data. We also introduce a methodology for interpreting the learned representation from our model that extracts time-sensitive insights about the process of passing or failing a given online course. Our experimental results demonstrate that student state representations learned by our model have better pre-

dictive power on the task of determining student failure rate than a baseline that only uses click stream data. We also present how individual topic-based insights into the process of passing or failing a course let us construct a rich characterization of a student likely to fail or pass an online course.

7. ACKNOWLEDGEMENT

This work was funded in part by NSF grant IIS 1822831.

References

- [1] Alison Ashby. Monitoring student retention in the open university: definition, measurement, interpretation and action. *Journal of Open, Distance and e-Learning*, 2004.
- [2] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *JMLR*, 2003.
- [3] Nigel Bosch, Yuxuan Chen, and Sidney D’Mello. It’s written on your face: detecting affective states from facial expressions while learning computer programming. In *ICITS*, 2014.
- [4] Anthony F Botelho, Ryan S Baker, and Neil T Heffernan. Improving sensor-free affect detection using deep learning. In *ICAIED*, 2017.
- [5] Anthony F. Botelho, Ryan Shaun Joazeiro de Baker, Jaclyn Ocumpaugh, and Neil T. Heffernan. Studying affect dynamics and chronometry using sensor-free detectors. In *EDM*, 2018.
- [6] Carol Elaine Calvert. Developing a model and applications for probabilities of student success: a case study of predictive analytics. *Journal of Open, Distance and e-Learning*, 2014.
- [7] Rafael A Calvo and Sidney D’Mello. Affect detection: An interdisciplinary review of models, methods, and their applications. *IEEE Transactions on affective computing*, 2010.
- [8] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv:1409.1259*, 2014.
- [9] Sidney D’mello and Arthur Graesser. Mind and body: Dialogue and posture for affect detection in learning environments. *Frontiers in Artificial Intelligence and Applications*, 2007.
- [10] Michael Eagle, Ted Carmichael, Jessica Stokes, Mary Jean Blink, John Stamper, and Jason Levin. Predictive student modeling for interventions in online classes. In *EDM*, 2018.
- [11] Mi Fei and Dit-Yan Yeung. Temporal models for predicting student dropout in massive open online courses. In *ICDMW 2015*.
- [12] Neil T Heffernan and Cristina Lindquist Heffernan. The assistants ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *IJAIED*, 2014.
- [13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.
- [14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.
- [15] Changchun Liu, Pramila Rani, and Nilanjan Sarkar. An empirical study of machine learning techniques for affect recognition in human-robot interaction. In *ICIRS*, 2005.
- [16] Julia M Matuga. Self-regulation, goal orientation, and academic achievement of secondary students in online university courses. *Journal of Educational Technology & Society*, 2009.
- [17] Mihalis A Nicolaou, Hatice Gunes, and Maja Pantic. Continuous prediction of spontaneous affect from multiple cues and modalities in valence-arousal space. *IEEE Transactions on Affective Computing*, 2011.
- [18] Zachary A Pardos, Ryan SJD Baker, Maria OCZ San Pedro, Sujith M Gowda, and Supreeth M Gowda. Affective states and state tests: Investigating how affect and engagement during the school year predict end-of-year learning outcomes. *Journal of Learning Analytics*, 2014.
- [19] Bart Rienties, Avinash Borooow, Simon Cross, Chris Kubiak, Kevin Mayles, and Sam Murphy. Analytics4action evaluation framework: A review of evidence-based learning analytics interventions at the open university uk. *Journal of Interactive Media in Education*, 2016.
- [20] Vernon C Smith, Adam Lange, and Daniel R Huston. Predictive modeling to forecast student outcomes and drive effective interventions in online community college courses. *Journal of Asynchronous Learning Networks*, 2012.
- [21] Steven Tang, Joshua C Peterson, and Zachary A Pardos. Deep neural networks and how they apply to sequential education data. In *L@S*, 2016.
- [22] Wei Wang, Han Yu, and Chunyan Miao. Deep model for dropout prediction in moocs. In *ICCSE 2017*.

Collaborative Problem-Solving Process in A Science Serious Game: Exploring Group Action Similarity Trajectory

Jina Kang
Utah State University
Logan, UT 84322
jina.kang@usu.edu

Dongwook An
University of Texas at Austin
Austin, TX 78712
louis.dongwook.an@utexas.edu

Lili Yan
Utah State University
Logan, UT 84322
liliyan@aggiemail.usu.edu

Min Liu
University of Texas at Austin
Austin, TX 78712
mliu@austin.utexas.edu

ABSTRACT

Collaborative problem-solving (CPS) as a key competency required in the 21st century. There has been an increasing need to understand CPS since it involves not only cognitive but also social processes, and thus its process is difficult to examine. Recent research has highlighted that computer-based learning environments provide an opportunity for students to collaborate with others to solve scientific problems and facilitate their knowledge building process, which can be dynamically tracked within the systems. However, limited research has attempted to identify CPS process captured in the computer-based learning environments designed for supporting CPS. This study therefore aimed to investigate students' CPS process in a serious game, *Alien Rescue*, by analyzing a student's daily tool use action sequence generated in the game. First, we computed a daily gameplay action similarity among students in a group using a similarity coefficient, *Jaccard* (*Jac*). Each group's *Jac* coefficients over the entire gameplay period (i.e. six days over three weeks) were considered as the group action similarity trajectory. The *Jac* coefficient of each day was entered as a single feature (i.e. a total of six features) to conduct a *KmL* cluster analysis that clusters longitudinal data. Three clusters of groups with similar behavior traits (i.e. group action similarity trajectories) were identified. The groups' background information (e.g. solution scores, knowledge gain scores) further provided how the groups' CPS traits can be related to their learning performance.

Keywords

Collaborative problem-solving, learning process, serious game, *Jaccard* coefficient, *KmL* cluster analysis, science learning.

1. INTRODUCTION

Research has highlighted a need for a comprehensive understanding of collaborative problem-solving (CPS), which is

regarded as one of the critical competencies of the 21st century skills [10]. OECD [21] recently defined CPS competency as "the capacity of an individual to effectively engage in a process whereby two or more agents attempt to solve a problem by sharing the understanding and effort required to come to a solution and pooling their knowledge, skills and efforts to reach that solution." Computer-based learning environments offer opportunities to monitor collaborative process in order to engage learners in building a shared understanding of a complex problem and support them in knowledge construction process by providing prompts to respond to their learning process or triggering real-time interventions to improve their CPS process [23, 25]. The captured log data including temporal and spatial students' behaviors within the system can reveal emergent patterns that not only reflect individual and group behaviors during CPS activities within the system, but also engage groups with diverse behavior patterns in an effective CPS process accordingly. Despite of these benefits, scanty research has attempted to examine collaboration process captured within a computer-based learning environment designed for supporting CPS. In addition, research on CPS has been mostly conducted by investigating verbal communications (e.g. [13]).

To address this gap, this study aims to investigate groups' CPS process in a serious game, *Alien Rescue*, using an individual student's daily tool use action sequence generated during the entire gameplay period (i.e. a total of six days over three weeks). We used a similarity coefficient, *Jaccard* (*Jac*), to calculate a daily gameplay action similarity among students in a group. *Jac* coefficients of each group over the entire gameplay period, serving as the group's action similarity trajectory, were used to identify patterns of group action similarity trajectory. The findings provide empirical evidences of diverse patterns of CPS process emerged as students engaged in their CPS activities in the serious game. Further, we discuss design considerations of serious games and how our application of the methods can be applied to future studies.

2. RELEVANT WORK

2.1 Collaborative Problem-Solving Process

CPS process has become a field of interest among researchers with the potential to get a better understanding of CPS activities. A review of recent research revealed that CPS phases and synchrony were two topics central to the research on CPS process. Researchers have identified several phases of CPS activities. Informed by

Jina Kang, Dongwook An, Lili Yan and Min Liu "Collaborative problem-solving process in a science serious game: Exploring Group Action Similarity Trajectory" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 336 - 341

dynamical systems approaches (e.g. [1]), researchers [29] investigated problem-solving phase transition by identifying an entropy peak in transition between the phases including knowledge construction, group problem model, group consensus, and evaluation (see more details in [6]). The entropy peak corresponds to shifts in communication in a problem-solving process within a group. The results showed empirical evidences that groups exhibit phase transitions during their CPS. The concept of initiative has been considered as another way of investigating CPS phases. Howard, Di Eugenio, Jordan, and Katz [11] examined *task initiative* shifts during CPS, which is one type of initiatives and refers to the participation of people in a conversation and their contributions to problem-solving activities during the conversation. They found that group members took *task initiative* when attempting at adding new contents which help the members to advance problem-solving.

Synchrony is another critical factor in understanding CPS process. Synchrony can be developed as group members reach out a shared problem space, which leads to potential transformation or advancement in their problem-solving process. Mercier and Higgins [18] elaborated on the concept of “a joint problem space” [20] and highlighted the importance of a joint understanding of the problem, which can be successfully developed when group members all come to understand the problem that has been worked on. The effectiveness of group work is related to the convergence of the individual members’ mental models [4]. Cukurova, Luckin, Millán, and Mavrikis [3] illustrated the significance of synchrony among group members. They provided the evidence of a positive relationship between CPS competence and member synchrony; that is, high competence CPS groups tended to have high levels of member synchrony.

2.2 Similarity Coefficients in Serious Games Analytics

A similarity measure is a statistical method to determine how (dis) similar one object is from the other ones by quantifying the similarity or distance between the objects. Typically, the objects being compared can be text strings, audios, images, videos, and navigation sequences, etc. Mathematically, a similarity metric is measured within the range of 0 to 1, indicating two objects are identical (1) and completely different (0). Various applications of similarity measures have been applied in emerging fields of technology, such as audio match and facial recognition. There are five most commonly used similarity measures, namely, Dice, Jaccard (*Jac*), Overlap, Cosine, and the Longest Common Substring coefficients (see details in [15]). In this paper, we used *Jac* coefficient. The use of *n*-gram is an indispensable step to calculate a similarity coefficient, when the directionality or contexts between objects is an important concern [15]. By using *n*-gram, researchers are able to set the sequence of objects before the calculation of similarity coefficients. The *n*-grams are named by the size of the sliding windows used—hence, unigram ($n = 1$), bigram ($n = 2$), trigram ($n = 3$), and fourgram ($n = 4$) and so on.

A serious game has shown its support to improve learners’ CPS performance with the chance to develop problem-solving and collaboration skills and with higher learning motivation (e.g. [25]). As players’ actions and behaviors in serious games are considered as the evidence in understanding CPS processes, researchers take serious games as the tool to observe and infer the players’ decision-

making process (e.g. [15]). Similarity coefficients have been applied to investigate the players’ gaming process. Osborn and Mateas [22] defined a (dis) similarity metric for the comparison of players’ sequences of actions. They found that the tool *Gamalyzer* (an exploratory visualization of gameplay traces) with the proposed (dis)similarity metric is valid in visualizing the overall strategies of game players. Learning performance in serious games can be quantified with the application of similarity measures to compare the course of action between novice and expert players (e.g. [16]). Loh et al. [15] examined several commonly used similarity measures to determine which measure or combination of measure would be viable in differentiating novice from expert players in serious games. Their findings showed that combining different similarity measures showed stronger predicting abilities than using a single similarity.

3. METHODS

3.1 Participants

The participants included sixth graders ($n = 196$) from a middle school in the Southwestern area of the United States. The participants played a serious game, *Alien Rescue*, as a part of science curriculum over three weeks. The teachers encouraged students to group between 2-4 students, but also allowed students to work individually during the gameplay period. There was a total of 70 groups. Each student in a group used their own laptop and solved the problems in collaboration with group members by collecting required information and eliminating planets or moons to find out the most suitable homes for each alien species. In order to investigate students’ collaborative problem-solving process, we only included students who worked in a group ($n = 156$). The students were balanced in terms of gender (77 males and 79 females). At-risk¹ students comprised 51.3% of the sample.

3.2 Serious game

Alien Rescue (<http://alienrescue.edb.utexas.edu>) is an open-ended serious game that allows students to discover multiple pathways to solve a problem [9]. In this game, students play in the role of young scientists who are asked to join the United Nations in the effort to rescue six alien species displaced from different places in a distant galaxy by helping them to find new homes in our solar system. Students are engaged in scientific investigations without explicit guidance in their problem-solving process. Students are able to develop a mastery by trying out multiple ways of solving the problem, such in finding evidence, matching information, and formulating rationales. Students develop high-level cognitive skills (i.e. goal setting, hypothesis generation, problem-solving, and self-regulation) while exploring the game environment. The previous studies (e.g. [12, 14]) showed empirical evidences of problem-solving stages within the game; that is, initial exploring and problem identification, background research including gathering and integrating information, hypothesis generation and testing, and solution generation. A set of cognitive tools are provided in the game to support students’ problem-solving process (see more details of each tool in [14]). Students are challenged to identify relevant information of the solar system by using in-game cognitive tools and match the information with each alien’s needs and characteristics. To solve the complex and ill-structured problem, students need to use the tools strategically. Students get access to the cognitive tools through a two-layer interface. Tools in first layer

¹ The school identified students as being at-risk of dropping out of a school by the state-defined criteria including low-performance

on an assessment instrument and limited English proficiency [27].

can be accessed one at a time while the six tools in the second layer can be used anytime overlaid with other tools (see Figure 1).

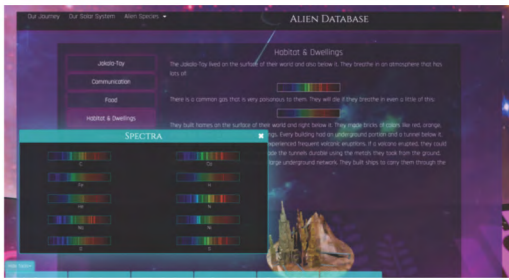


Figure 1. Alien Database overlaid with Spectra

3.3 Data sources

3.3.1 Performance scores

Before and after gameplay, an individual student's comprehension of factual and applied scientific knowledge introduced in the game was measured using a Space Science Knowledge Test (SSKT). SSKT consists of twenty-four multiple choice items (Cronbach's $\alpha = 0.77$), which score ranges between 0 and 24 (1 point for each item) logged in the system. An individual student's SSKT gain score was calculated by subtracting the pretest score from the posttest score. Then, each group's average gain score was calculated (i.e. total gain scores / a number of students in a group), which is considered as each group's after-game performance.

In addition, the game logs a student's written recommendation(s) for each alien, in which they must indicate an appropriate home for each species and provide a rationale. Students can submit multiple recommendations for each alien species, which reveals the results of students' problem-solving processes—that is, justifications of their solutions using the gathered data during the gameplay. The solutions were evaluated using an 8-point rubric used in previous studies (see more details in [2]) in terms of the correctness of the solution and the number of reasons to the selected home. Each group's average solution score is considered as the group's in-game performance.

3.3.2 Gameplay data

The gameplay data—that is, the user-generated data derived directly from students' actions within the game—were used to identify students' navigation patterns as they engaged in *Alien Rescue*. The game logs every action as each student interacts with the environment. The gameplay data contains a student identifier, a cognitive tool that the student accessed, a type of action (e.g., open, close, click), an additional note on student's interactions, and a timestamp for each action (see an example of data in Table 1). "Open" indicates a student opens a tool, while "Click" indicates a student clicks a submenu of the tool.

Table 1. Example of A Student's Navigation Data

Tool	Action	Notes	Timestamp
Probe Design	Open		5/17 10:33:19
Solar System	Click system	Mercury	5/17 10:36:48
Concepts	Open		5/17 10:48:00
Concepts	Close		5/17 10:49:06

3.4 Analysis

3.4.1 Group action similarity using a Jaccard coefficient

We computed a gameplay action similarity between students in a group with a *Jac* coefficient (see 2.2). In order to calculate the similarity of students' navigation traces for each day, we cleaned and transformed each student's navigation data into a 'bag of words.' For example, assuming that on Day1, one student's navigation is represented by *string A* = "Probe Design Open, Solar System Click Mercury, Concepts Open, ...", and the other student's data is expressed by *string B* = "Solar System Open, Solar System Click Mercury, Solar System Click Venus, ...", we can obtain the intersection set of *A* and *B* ($A \cap B$) and union set of *A* and *B* ($A \cup B$). The *Jac* coefficient is therefore the length of intersection set over the length of the union set. In order to check if there is a directionality between students' actions, we also applied a bigram setting to the navigation sequence and calculated a *Jac* coefficient. A bigram sequence was obtained using a 'sliding window' of size 2. We conducted a descriptive analysis to compare the distributions of the unigram with bigram *Jac* coefficients of all groups. As shown in Figure 2, the groups' unigram *Jac* coefficients were overall normally distributed, while the bigram *Jac* coefficients showed highly skewed to zero (i.e. small variance). Therefore, we decided to use a unigram *Jac* coefficient for this study. As Loh et al. [15] suggested using a larger *n*-gram, when any contextual relationship between actions are critical, we included a type of action (see 3.3.2) and split the data into each day to further consider the context and directionality.

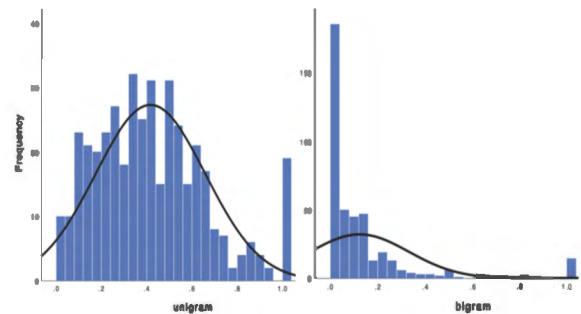


Figure 2. Histograms of unigram and bigram *Jac* coefficients

Note. Unigram ($M = .42$, $SD = .24$), Bigram ($M = .12$, $SD = .21$)

3.4.2 Clustering analysis for longitudinal data

A *Jac* coefficient of each day was entered as a single feature (i.e. a total of six features) to conduct a cluster analysis to identify the potential clusters of collaborative groups with similar behavior traits (i.e. group action similarity trajectory). Six *Jac* coefficients of one group can be seen as the action similarity trajectory of the group over the gameplay. To handle such trajectory data, we used a *KmL* package in R, which is a new implementation of *k*-means designed to analyze longitudinal data [8]. One common problem in longitudinal studies is missing data (e.g. [17]). While *k*-means is unable to handle missing values and normally excludes missing data, *KmL* provides diverse imputation methods (e.g. linear interpolation, copyMean; see more details [7]) to deal with different types of missing values including intermittent missing data (data missing in the middle of a trajectory) and monotone missing data (data missing either at the beginning or end) [19]. We were unable to calculate a coefficient when there was only one student in a group logged in the game on a certain day. There were such missing

values found randomly over six gameplay days. Therefore, we used linear interpolation (Bisector) to handle missing values of the *Jac* dataset, since the method considers not only a local intermediate line (not just sensitive to first or last values), but also a bisector between a global and local lines.

KmL provides methods to define starting conditions and an optimal number of clusters and an easy way to run several *k*-means. *KmL* transforms longitudinal data into an object called ‘ClusterLongData.’ Once the object has been created, *KmL* runs *k*-means several times and stores all the clusters that the algorithm finds over each iteration of finding an optimal partition in the object. *KmL* also offers a tool that can visualize partitions, in which researchers can make a decision on the best partition by comparing different criteria including Calinski & Harabatz, Ray & Turi, and Davies & Bouldin (e.g. [7]). In this study, 3 clusters were suggested as the optimal number of clusters.

The data failed the major assumption of the one-way ANOVA (i.e. the non-normally distribution assumption). We thus conducted a non-parametric test, Kruskal-Wallis H test, to confirm a statistical significance of the group action similarity trajectories between clusters since a cluster analysis can only reveal the latent cluster patterns. In addition, the cluster patterns were visualized for deeper understanding of group action similarity trajectories in each cluster.

4. RESULTS

As shown in Table 2, the average values of daily *Jac* coefficients as exhibited by the three clusters of groups achieved the level of significance (χ^2), indicating the groups were well-partitioned into each cluster. Kruskal-Wallis H tests overall showed that there were statistically significant differences between the mean ranks of daily *Jac* coefficients at least in one pair of clusters. Dunn’s pairwise tests for each *Jac* coefficient were carried out for the three pairs of clusters (i.e. Cluster 1 & Cluster 2, Cluster 2 & Cluster 3, and Cluster 1 & Cluster 3). We further examined the background of the groups in each cluster including the average SSKT gain score and the number of groups who submitted at least one solution. To further investigate potential patterns between the clusters, line charts of action similarity trajectories grouped by each cluster including a similarity trajectory trend were derived (see Figure 3). As shown in Figure 3, the line charts indicated that the similarity trajectory trends of the groups in each cluster were distinctively different.

Approximately 40% of the groups are centered in Cluster 1, and the mean ranks of *Jac* coefficients were overall lower than those of the other two clusters. As shown in Figure 3, this cluster’s overall

similarity trajectory decreased slightly. This cluster showed the lowest solution submission rate. These groups achieved the lower average solution scores and SSKT gain scores than the other clusters.

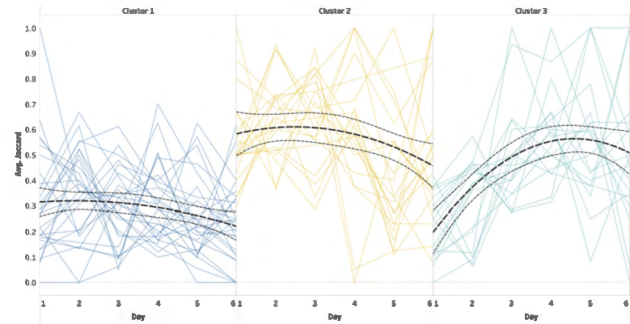


Figure 3. Similarity Trajectories of Groups in Each Cluster

Note. The dashed line shows a polynomial trend model of degree 2 computed for ‘Average of *Jac*’ given ‘Day’ (dotted lines for a confidence band).

About 30% of the groups are in Cluster 2, and the groups exhibited overall the highest mean ranks of *Jac* coefficients except for Day 5. Similar to Cluster 1, the groups in Cluster 2 exhibited a decreasing similarity trajectory trend toward the end of gameplay, but maintained high similarity in their group actions during the first few days. The groups in this cluster showed the highest solution submission rate. Both their average solution and SSKT gain scores showed that their average performance was close to the mean performance of all clusters.

Lastly, the rest of the groups are centered in Cluster 3. The groups in Cluster 3 exhibited an increasing similarity trajectory trend with the peak on Day 5. It is worth noting that their *Jac* coefficients during the first two days were recorded as the lowest mean ranks among all clusters. The solution submission rate of Cluster 3 is close to the average solution submission rate across all clusters (31.58%). However, the groups who submitted at least one solution performed better at their solution submissions than other clusters. Additionally, the groups in this cluster performed the best at their SSKT gain score.

Table 2. Cluster Membership Description

Cluster	Group SSKT Gain score (Mean)	^a No. of groups with solution(s)	<i>Jac</i> Day1	<i>Jac</i> Day2	<i>Jac</i> Day3	<i>Jac</i> Day4	<i>Jac</i> Day5	<i>Jac</i> Day6	<i>Jac</i> (Mean)
C1 (<i>n</i> = 29, 41.43%)	1.727	6 (20.69%, ^b <i>M</i> _{solution} = 2.00)	0.3057 ^d (30.12)	0.3398 (27.10)	0.2901 (20.26)	0.3079 (20.55)	0.2503 (23.98)	0.2256 (22.21)	0.2866
C2 (<i>n</i> = 22, 31.43%)	2.682	9 (40.91%, <i>M</i> _{solution} = 3.67)	0.5473 (53.84)	0.6427 (55.41)	0.6420 (52.55)	0.5982 (46.64)	0.4149 (38.02)	0.5170 (45.18)	0.5603
C3 (<i>n</i> = 19, 27.14%)	3.412	6 (31.58%, <i>M</i> _{solution} = 4.00)	0.2218 (22.47)	0.3213 (25.26)	0.4961 (39.03)	0.5785 (45.42)	0.5609 (50.16)	0.4963 (44.58)	0.4458
^c χ^2			27.73***	30.80***	32.29***	26.77***	19.49***	21.18***	

Note. ^aThe number of groups submitted at least one solution (Percentage of the groups in each cluster); ^bAverage of the total solution scores of groups in each cluster; ^cKruskal-Wallis H test results, ****p* < 0.001; ^dMean ranks

5. DISCUSSION

We applied a *Jac* coefficient to investigate CPS process within a serious game, *Alien Rescue*: that is, (1) to compute a daily gameplay action similarity among students in a group and (2) to identify action similarity trajectory patterns across groups using each group's similarity trajectory.

Although the groups in Clusters 1 and 2 exhibited overall a decreasing similarity trend, Cluster 1 showed relatively lower action similarities over the entire gameplay, indicating the group members did not use the same tools. The group members seemed to maintain the individual tool use tendency over time. Both Clusters 2 and 3 showed relatively higher *Jac* coefficients and higher in-game (i.e. solution score) and after-game (i.e. knowledge gain score) learning performances, which, together with the chi-square test, can be most likely seen there might be a potential positive relationship between a *Jac* coefficient and learning performance. However, the two clusters' *Jac* coefficients followed a different pattern; that is, the *Jac* coefficients of Cluster 3 have risen considerably over the gameplay period, and the group in Cluster 3 started off with the lowest action similarity among all clusters. Additionally, the results of Dunn's pairwise tests showed the *Jac* coefficients were significantly different ($p < .001$) between Cluster 2 and Cluster 3 only during the first two days. Research has shown the importance of the convergence of individual's mental models in CPS [4] and the positive association between CPS competence and a level of member synchrony [3]. The findings therefore suggest that, during the early gameplay days, the group members in Cluster 3 came to successfully understand the problem and engaged in their collective cognitive process. This further supports the fact that a group action similarity trajectory can be an indicator of the process of developing shared problem space between group members [18].

We applied *n*-gram to compute a *Jac* similarity coefficient: unigram and bigram. Compared with the groups' unigram *Jac* coefficients, the bigram *Jac* coefficients showed a small variance (i.e. highly skewed to zero). In this study, a unigram *Jac* coefficient is therefore a viable way in understanding different levels of group collaboration in this serious game. Research on serious games analytics highlighted the use of *n*-gram would be critical to understand directionality and contexts between events or actions [15, 28]. Since a unigram can possibly ignore the context and directionality of actions, we included a type of action (i.e. click a sub-menu in each tool) to further consider the context and split the data into each day to preserve the directionality when computing a *Jac* coefficient. Such modification is needed when applying *n*-grams to different purposes of study. In addition, we applied a bigram to further examine the frequent sequences of groups in each cluster.

KmL, a cluster analysis for longitudinal data, has been often used in scientific disciplines such in medical research [7, 8]. The *KmL* clustering results in this current study showed remarkable differences of the groups' action similarity trajectories among three clusters, which indicate different patterns of CPS process in the serious game. In particular, the positive action similarity growth of Cluster 3 demonstrated that they developed a shared understanding of the problem during the early gameplay days, which has been considered as a critical process of successful collaboration in CPS activities [5, 23]. It is confirmed by the fact that the learning performance of group members of Cluster 3 was higher than that of the groups in other clusters, indicating their experience throughout the CPS process was successfully transformed to their knowledge

gain. The findings highlight the importance of providing guidance for students who tend to work independently (i.e. Cluster 1) or who may simply replicate actions of other students in the group (i.e. Cluster 2) to engage in the process of developing a shared problem space. The results further inform design considerations of serious games that support CPS: for example, providing prompts with explicit inquiries, in which a group can be engaged in the successful CPS process grounded on the group's achievement of a shared understanding of a given problem. Taken together, this study confirms *KmL* as a promising method to examine features at different time points generated from gameplay data, which can be seen as an action trajectory that provides insights into CPS process in serious games.

Our work has limitations that should be addressed in future studies. First, the dataset is small and was collected at one middle school with little diversity; for example, 51.3% of the sample was labeled as at-risk, which may not be applicable in other schools with different settings. Second, understanding CPS process is critical, but challenging particularly in an open-ended learning environment like *Alien Rescue*. This work therefore should be expanded to include additional data such as video or audio recordings to capture group conversations and actions to provide robust evidence for the findings of this study. Third, our method of clustering group action trajectory patterns using the *KmL* clustering together with a *Jac* coefficient showed promising evidences to understand students' CPS process. However, due to the small sample size, this may need to be further explored at a larger scale. We are currently employing integrated analytical methods to better understand CPS process using such as mixture latent growth curve model to compare the cluster memberships with the results from *KmL*, and multilevel modeling to examine the relative influence of teacher (i.e. two teachers in the middle school) on the action similarity trajectories of the groups.

6. CONCLUSION

This study used a student's daily tool use action sequence generated in a serious game, *Alien Rescue*, to investigate the students' CPS process. We applied a similarity coefficient, *Jac*, to identify a group action similarity trajectory. The *KmL* clustering analysis discovered unique clusters of groups with similar group action trajectories, the membership of which further provided how CPS traits can be related to their learning performance. Each cluster's characteristics shed light on deriving design considerations to promote students' positive collaboration experience during CPS activities within serious games, and to engage teachers in facilitating students' effective CPS process. Lastly, the advantages and limitations of the methods employed in this study point toward the need for continued research on exploring potential analytical methods and scaling up the sample size to include more diverse population.

7. REFERENCES

- [1] Angus, D., Smith, A. E., and Wiles, J. 2012. Human communication as coupled time series: Quantifying multi-participant recurrence. *IEEE Transactions on Audio, Speech, and Language Processing*, 20, 6 (Aug. 2012), 1795-1807. DOI= <https://doi.org/10.1109/TASL.2012.2189566>.
- [2] Bogard, T., Liu, M., and Chiang, Y. H. 2013. Thresholds of knowledge development in complex problem solving: A multiple-case study of advanced learners' cognitive processes. *Etr. & Educ. Tech. Res.* 61,3 (Jun. 2013), 465-503. DOI= <https://doi.org/10.1007/s11423-013-9295-4>.

- [3] Cukurova, M., Luckin, R., Millán, E., and Mavrikis, M. 2018. The NISPI framework: Analysing collaborative problem-solving from students' physical interactions. *Computers & Education*, 116 (Jan 2018), 93-109. DOI= <https://doi.org/10.1016/j.compedu.2017.08.007>.
- [4] DeFranco, J. F., Neill, C. J., and Clariana, R. B. 2011. A cognitive collaborative model to improve performance in engineering teams—A study of team outcomes and mental model sharing. *Syst. Eng.* 14, 3 (Oct. 2011), 267-278. DOI= <https://doi.org/10.1002/sys.20178>.
- [5] Dillenbourg, P. 1999. What do you mean by collaborative learning?. P. Dillenbourg. *Collaborative-learning: Cognitive and Computational Approaches.*, Oxford: Elsevier, pp.1-19.
- [6] Fiore, S. M., Rosen, M. A., Smith-Jentsch, K. A., Salas, E., Letsky, M., and Warner, N. 2010. Toward an understanding of macrocognition in teams: Predicting processes in complex collaborative contexts. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 52, 2 (Jul. 2010), 203-224. DOI= <https://doi.org/10.1177/0018720810369807>.
- [7] Genolini, C., Alacoque, X., Sentenac, M., and Arnaud, C. 2015. kml and kml3d: R packages to cluster longitudinal data. *J. Stat. Softw.* 65(4), 1-34. DOI= <https://doi.org/10.18637/jss.v065.i04>.
- [8] Genolini, C., and Falissard, B. 2011. KmL: A package to cluster longitudinal data. *Comput. Meth. Prog. Bio.* 104, 3 (Dec. 2011), 112-121. DOI= <https://doi.org/10.1016/j.cmpb.2011.05.008>.
- [9] Glaser, R. 1991. The maturing of the relationship between the science of learning and cognition and educational practice. *Learning and Instruction*, 1, 2 (1991), 129-144. DOI= [https://doi.org/10.1016/0959-4752\(91\)90023-2](https://doi.org/10.1016/0959-4752(91)90023-2).
- [10] Griffin, P., McGaw, B., and Care, E. 2012. *Assessment and teaching of 21st century skills: Methods and Approach*. New York, NY: Springer. DOI= <https://doi.org/10.1007/978-94-007-2324-5>.
- [11] Howard, C., Di Eugenio, B., Jordan, P., and Katz, S. 2017. Exploring initiative as a signal of knowledge co-construction during collaborative problem solving. *Cognitive sci.* 41, 6 (Nov. 2017), 1422-1449. DOI= <https://doi.org/10.1111/cogs.12415>.
- [12] Kang, J., Liu, M., and Qu, W. 2017. Using gameplay data to examine learning behavior patterns in a serious game. *Comput. Hum. Behav.* 72 (Jul. 2017), 757-770. DOI= <https://doi.org/10.1016/j.chb.2016.09.062>.
- [13] Keyton, J., Beck, S. J., and Asbury, M. B. 2010. Macrocognition: a communication perspective. *Theoretical Issues in Ergonomics Science*, 11, 4 (Jun 2010), 272-286. DOI= <https://doi.org/10.1080/14639221003729136>.
- [14] Liu, M., and Bera, S. 2005. An analysis of cognitive tool use patterns in a hypermedia learning environment. *Educational Technology Research and Development*, 53, 1 (Mar. 2005), 5-21. DOI= <https://doi.org/10.1007/BF02504854>.
- [15] Loh, C. S., Li, I. H., and Sheng, Y. 2016. Comparison of similarity measures to differentiate players' actions and decision-making profiles in serious games analytics. *Comput. Hum. Behav.* 64, 562-574. DOI= <https://doi.org/10.1016/j.chb.2016.07.024>.
- [16] Loh, C. S., and Sheng, Y. 2015. Measuring the (dis-)similarity between expert and novice behaviors as serious games analytics. *Education and Information Technologies*, 20, 1 (Nov. 2015), 5-19. DOI= <http://doi.org/10.1007/s10639-013-9263-y>.
- [17] Mallinckrod, C. H., Lane, P. W., Schnell, D., Peng, Y., and Mancuso, J. P. 2008. Recommendations for the primary analysis of continuous endpoints in longitudinal clinical trials. *Drug Inf. J.* 42, 4 (Jul.2008), 303-319. DOI= <https://doi.org/10.1177/009286150804200402>.
- [18] Mercier, E., and Higgins, S. 2014. Creating joint representations of collaborative problem solving with multi-touch technology. *J. Comput. Assist. Lear.* 30, 6 (Feb. 2014), 497-510. DOI= <https://doi.org/10.1111/jcal.12052>.
- [19] Molenberghs, G., Thijs, H., Jansen, I., Beunckens, C., Kenward, M. G., Mallinckrodt, C., and Carroll, R. J. 2004. Analyzing incomplete longitudinal clinical trial data. *Biostatistics*, 5, 3 (Jul. 2004), 445-464. DOI= <https://doi.org/10.1093/biostatistics/kxh001>.
- [20] Newell, A., and Simon, H. A. 1972. *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.
- [21] Organisation for Economic Co-operation and Development. (2017). PISA 2015: Draft collaborative problem solving framework. Retrieved from <https://www.oecd.org/pisa/pisaproducts/Draft%20PISA%202015%20Collaborative%20Problem%20Solving%20Framework%20.pdf>.
- [22] Osborn, J. C., and Mateas, M. 2014. A game-independent play trace dissimilarity metric. In FDG (April).
- [23] Roschelle, J., and Teasley, S. D. 1995. The construction of shared knowledge in collaborative problem solving. In *Computer supported collaborative learning* (pp. 69-97). Springer, Berlin, Heidelberg.
- [24] Saab, N., van Joolingen, W., and van Hout-Wolters, B. 2012. Support of the collaborative inquiry learning process: Influence of support on task and team regulation. *Metacog. Learn.* 7, 1 (Mar. 2012), 7-23. DOI= <https://doi.org/10.1007/s11409-011-9068-6>.
- [25] Sánchez, J., and Olivares, R. 2011. Problem solving and collaboration using mobile serious games. *Computers & Education*, 57, 3 (Nov. 2011), 1943-1952. DOI= <https://doi.org/10.1016/j.compedu.2011.04.012>.
- [26] Stewart, A. E., Keirn, Z. A., and D'Mello, S. K. 2018 October. Multimodal Modeling of Coordination and Coregulation Patterns in Speech Rate during Triadic Collaborative Problem Solving. In *Proceedings of the 2018 on International Conference on Multimodal Interaction* (pp. 21-30). ACM. DOI= <https://doi.org/10.1145/3242969.3242989>.
- [27] Texas Education Agency. (2017).
- [28] van der Loo, M. P. 2014. The stringdist package for approximate string matching. *The R Journal*, 6, 1 (2014), 111-122. <https://doi.org/10.32614/RJ-2014-011>.
- [29] Wiltshire, T. J., Butner, J. E., and Fiore, S. M. 2018. Problem-Solving Phase Transitions During Team Collaboration. *Cognitive Sci.*, 42, 1(Jan. 2018), 129-167. DOI= <https://doi.org/10.1111/cogs.12482>.

Generalizing Expert Misconception Diagnoses Through Common Wrong Answer Embedding

John Kolb
UC Berkeley
jkolb@berkeley.edu

Scott Farrar^{*}
Khan Academy / Independent
scottfarrar@gmail.com

Zachary A. Pardos
UC Berkeley
pardos@berkeley.edu

ABSTRACT

Misconceptions have been an important area of study in STEM education towards improving our understanding of learners' construction of knowledge. The advent of large-scale tutoring systems has given rise to an abundance of data in the form of learner question-answer logs in which signatures of misconceptions can be mined. In this work, we explore the extent to which collected expert misconception diagnoses can be generalized to held-out questions to add misconception semantics. We attempt this generalization by way of a question-answer neural embedding trained on chronological sequences of learner answers. As part of our study, we collect natural language misconception diagnoses from math educators for a sampling of student answers to questions within four topics on Khan Academy. Drawing inspiration from machine translation, we use a multinomial logistic regression model to explore how well the expert misconception semantics, in the form of bag-of-words vectors, can be mapped onto the learned embedding space and interpolated. We evaluate the ability of the space to generalize expert diagnoses using three levels of cross-fold validation in which we measure the recall of predicted natural language diagnoses across rater, topics, and questions. We find that the embedding provides generalization performance substantially beyond baseline approaches.

1. INTRODUCTION

The notion of mapping out abstract spaces of student learning and development has been around for ages, with Zone of Proximal Development [23] serving as a canonical example of defining the area of topics a student could learn with help from peers and the topics beyond. Work in Educational Data Mining has explored mapping out learning spaces taking the form of tree structures [4] or concept nodes in a directed graph [11], often used to represent prerequisite relationships. Other work has mapped out progress points within a course and their relationship to classical psychometric measures of

ability [1]. In this work, we build on the idea of conceiving a space of learning as an embedding, or set of continuous vectors, with parts of the space indicative of different states of understanding and misconception [14]. We learn this embedding from sequences of millions of answers to exercises from a popular STEM tutoring system, then recruit qualified experts to diagnose a sampling of common wrong answers, providing natural language semantics to associate with question answers at their respective locations in the embedding. To test if the embedding generalizes these short form diagnoses, we use linear interpolation of the learned vector space to predict the words used in held-out diagnoses, holding out by expert, problem type, and question in cross-validation experiments. Successful predictive generalization in this task has implications for surfacing automatically generated misconception hypotheses to both teachers and computer tutors.

2. RELATED WORK

The theory of mathematical misconceptions described by Piaget [16], and considered by Smith, diSessa, and Roschelle [19] is one of continually developing partial understandings. Analysis of learner responses, rather than only correctness, may reveal aspects of their understandings. In the age of big data and computation, several modern approaches have brought different perspectives to the analysis of misconceptions. Feldman et al. [5] generated plausible production rules that could have produced the common wrong answers observed in student responses to addition questions in 11 elementary schools. In the vein of KC model or Q-matrix improvement [22], Liu, Patel, & Koedinger [6] explored adding KCs symbolizing buggy production rules to problem steps whose correct answer could be arrived at in spite of applying the buggy production. They found that the inclusion of this item-level misconception tagging improved the overall fit of their AFM model and the validity of the learned individual student parameters. Most complimentary to our work is the work of Michalenko, Lan, & Baraniuk [8], who did not study misconceptions in common wrong answers, but rather misconceptions found in the text of long open response text, using skip-grams and other embedding methods. Their approach is complementary to ours in that it cannot be applied to short, numeric answers in isolation. Inversely, our approach, which extends the embedding context across questions, is driven by questions that generate common wrong answers across students, which would exclude direct applicability to long answer response text.

^{*}At Khan Academy 2016-2017

2.1 Buggy Rules

In the cognitive theories underlying the design of intelligent tutoring systems [2], there are rules that produce correct and consistent answers, and efforts have been made towards cataloging collections of buggy rules that could instead produce incorrect answers. These buggy rules could represent misconceptions that students often have during the learning process [3]. This large collection of buggy rules is often referred to as a bug catalog [20]. As a student moves through a problem set, the bug catalog enables tutoring systems to tag, track, and respond to a path of answers the student provides.

Past research efforts to classify these buggy rules also include the manual labeling of misconceptions by experts [7], the exploration of cluster relationships between the wrong answers [15], and approaches that take into account the frequency of student misconceptions [21]. These efforts lay the foundation for automated approaches which utilize these buggy rules to generate targeted guidance messages specific to each incorrect answer [18].

2.2 Use of Skip-grams

Skip-gram models were originally applied to the embedding of words based on a large corpus of text (e.g. Wikipedia or a large archive of news articles). Once trained, the representational (hidden) layer of these models was shown to encode distributed concepts in the form of syntactic (e.g., bee is to bees as goose is to geese) as well as semantic relationships (e.g., Einstein is to scientist as Picasso is to painter) [10]. While conventionally applied to language in its debut, skip-grams have been applied to non-linguistic data from education. University courses were embedded from sequences of enrollments [13] to find course similarities outside of what could be inferred from catalog descriptions. Questions within the ASSISTments tutoring platform were embedded based on sequences in which problems were answered in order to predict the skill of untagged questions [12]. Skip-grams and other embedding models have been applied to standard natural language in educational contexts, such as the learning of vector representations of open response text and correlating vector representations with the presence or absence of hand coded misconceptions[8].

3. TUTOR DATA SET

Our dataset of anonymized student answer logs comes from Khan Academy, an online STEM tutoring platform. As described in our previous work [14], Khan Academy categorizes student responses by *exercise*, a broad skill similar to those seen in ASSISTments Skill Builder sets; by *problem type*, a problem template; and finally by *seed*, one of two hundred values per problem type which uniquely identifies a template instantiation. Each log entry also contains an anonymous user ID and timestamp, which we use to group and chronologically sort student answers for model training.

We used the same exercise selection process as in [14] to narrow our focus to exercises with sufficient data and concerning topics that would likely surface interesting misconceptions for educators to analyze and describe. This involved consulting a subject matter expert in mathematical education [17] and verifying the correctness of the log entries by forming a sample set of questions and manually accessing their respective web pages on Khan Academy. At the conclusion of this

filtering process, we identified four suitable exercises to use in our experiments:

1. “Surface Areas” (SA)
2. “Slope from an equation in slope intercept form” (SESI)
3. “Area of quadrilaterals and polygons” (AQP)
4. “Adding and subtracting fractions” (ASF)

Table 1 shows statistics for each exercise.

	SA	SESI	AQP	ASF
Problem Types	6	2	2	7
Seeds	38	20	50	40
Students	105,659	33,603	58,239	179,263
Unique Incorrect Answers	55,126	6,912	17,998	46,516
Total Incorrect Answers	619,045	112,390	298,356	873,916

Table 1: Descriptive statistics of exercises used to train the skip-gram models.

A second dataset was collected as part of this study, which consisted of natural language diagnoses of common wrong answers from our chosen exercises. These diagnoses were written by mathematics educators, with each diagnosis explaining the misconception that was potentially responsible for the incorrect answer. We collected misconception diagnosis labels using an online survey platform.¹ We describe the collection of these data in Section 4.2.

4. METHODOLOGY

In this section, we describe the techniques employed to complete three primary methodological tasks:

1. Generate learned question answer embeddings from student answer logs
2. Generate bag-of-words representations of the semantic data contained in educator diagnoses of the misconceptions associated with the incorrect student answers from (1.)
3. Compute a model that generalizes semantic diagnoses of wrong answers based on regression from the continuous vectors of (1.) to the semantic representations from (2.)

Figure 1 depicts the full data processing and machine learning pipeline that we implemented to complete these tasks, using both the answer event logs and the misconception diagnoses as inputs and outputting natural language diagnoses for held-out question answers.

4.1 Embedding Student Answers

As described in Section 2, machine learning models originally intended to model natural language have recently been applied to a number of other domains, including education. Motivated by the success of these efforts, we used a skip-gram neural network model to learn representations of student answers. A representation in our setting, or *embedding*, is a vector in a high-dimensional space that is learned by a skip-gram model. We use the same strategy as in [14] to encode each student answer in a token containing its *seed* and the

¹<https://qualtrics.com>

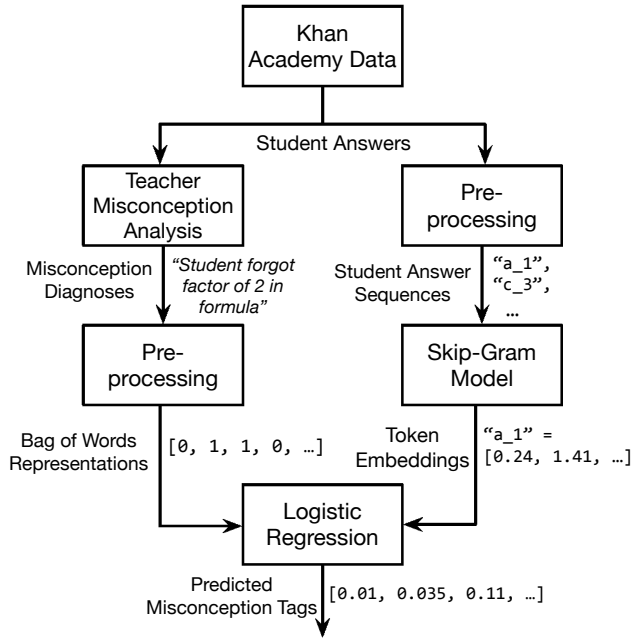


Figure 1: The pipeline used to model student answers, teacher diagnoses, and their correlation.

frequency rank of the student’s response within that seed. For example, if a student were to answer a question generated from seed x01b with the most frequently occurring incorrect answer to that question, their answer would be represented by the token x01b_1.

A skip-gram model is a two-layer neural network (one hidden layer) that analyzes a corpus of token sequences to learn continuous vector representations for each of these tokens. Vectors are trained with the goal of predicting the context of each token. For example, x01b_2 would have s03c_4 in its context if students often provide incorrect responses to those questions in succession. The loss function (Eq. 1) for the training process, described in [10], seeks to optimize the log-likelihood of the tokens in context given a specific input token. S represents the set of input sequences for the model, each corresponding to a student’s sequence of responses to a given exercise. c represents the window size, a hyperparameter of the model that specifies the width of a token’s context when learning its representation, and T represents the number of tokens in sequence s .

$$C = - \sum_{s \in S} \frac{1}{T} \sum_{t=1}^T \sum_{\substack{-c \leq j \leq c \\ j \neq 0}} \log P(w_{t+j} | w_t) \quad (1)$$

We use the negative sampling variant for training the skip-grams as introduced in [10], which replaces the final term of the form $\log P(w_O | w_I)$ in Equation 1 with

$$\log \sigma \left(v_{w_O}^T v_{w_I} \right) + \sum_{i=1}^k \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma \left(-v_{w_i}^T v_{w_I} \right) \right] \quad (2)$$

Above, σ represents the sigmoid function. Roughly, this formulation seeks to include the weights of k randomly chosen negative samples, i.e., tokens w_i that do *not* occur within

the context of the target token w_O , in the backpropagation process. Unlike the original hierarchical softmax formulation, negative sampling has the advantage of only adjusting pairs of weights in the underlying network during backpropagation.

4.2 Collecting Teacher Diagnoses

We collected expert-generated semantic misconception diagnosis data through a questionnaire designed and run on the Qualtrics platform. Qualtrics recruited survey participants and compensated them on our behalf at a rate of \$30 per participant. We had Qualtrics recruit participants who:

- Are working as a mathematics educator for students who are in grades 5–12 or undergraduates
- Have at least two years of prior teaching experience

The number of problem types and seeds within each exercise included in the survey is shown in Table 2. For each seed, we formed a batch of the five most frequently submitted incorrect answers to present to survey participants.

Exercise	# Prob. Types	# Seeds
Slope from an Equation in Slope Intercept Form	2	17
Adding and Subtracting Fractions	5	18
Surface Areas	6	36
Area of Quadrilaterals and Polygons	2	18

Table 2: Wrong answer exercises, problem types, and seeds for which expert diagnoses were sought

Each survey participant was provided with initial instructions, excerpted in Figure 2. Next, they were shown three randomly selected answer batches. For each batch, the survey respondent was presented with a screenshot of the original question as it appeared on Khan Academy, the text of the five incorrect student answers, and text boxes to write a brief misconception diagnosis for each answer. An example Khan Academy question and the associated diagnoses we collected are shown in Figure 3.

Respond with a general label-phrase that describes the most likely error or misconception related to the incorrect answer.

- Avoid references to specifics of the question (e.g., do not say “additive inverse is 4, not -4 ”).
- Your label or phrase should be general enough such that it could potentially be applied to other incorrect answers. Therefore, you may duplicate labels and phrases as you see appropriate.
- Avoid abbreviations (e.g., use “y intercept” instead of “yint”).

Example Responses

Question: Solve $3x - 4 = 20$

Student Answer: $5 \frac{1}{3}$

Example Label-Phrase: opposite of additive inverse

Figure 2: An excerpt of the instructions presented to survey participants providing expert diagnoses

Alternatively, we could have asked experts to create misconception labels out of terms drawn from a fixed taxonomy,

$$-\frac{1}{4} - \left(-\frac{3}{5}\right) = \square$$

Answer	Misconception Diagnosis
17/20	Added 5 + 12 instead of 5 − 12.
−17/20	Added 5 + 12 instead of 5 − 12. And used incorrect sign.
−7/20	Has incorrect sign. Should be +.
2/5	Did not use common denominator.

Figure 3: A sample Khan Academy question and corresponding misconception labels

rather than to compose these labels from scratch and without explicit guidance. However, the terms in this taxonomy would inevitably reflect our own biases and assumptions and may prevent experts from accurately describing their observations. Instead, we allowed a broad vernacular, but also asked experts to review their labels at the end of the survey to encourage them to be consistent in their language.

We found that the quality of survey responses varied dramatically within our dataset and developed a procedure to identify and retain only misconception labels that were suitable for further analysis. We manually excluded all responses where an attempt at a label was clearly not present, such as “idk.” Next, we retained diagnoses only from experts who wrote labels with an average length of 20 characters or more. This process left us with 570 unique diagnoses covering 14 of the 15 problem types and 64 of the 89 seeds.

4.3 Processing Teacher Diagnoses

After collecting expert misconception diagnoses through the survey platform, we performed data pre-processing to eventually represent each label in bag-of-words form. Many diagnoses contained references to specific numbers found in the instantiation of the question. We chose not to give every numerical quantity its own token but rather to replace each contiguous mathematical expression with the token `numN`, representing the N^{th} contiguous expression appearing in the diagnoses for each seed. Numbers used to describe general misconception rules, e.g. the factor of $1/2$ used in computing the area of a triangle, were hand-identified and allowed to be represented in original form. This helps to prevent our models from incorrectly identifying correlations that are coincidental (two question instances happen to use the same random quantity) rather than structural.

Next, we stripped punctuation, removed stopwords, and performed word stemming. Finally, we manually removed some of the most common tokens that we deemed uninformative and which could have resulted in trivially easy prediction due to their frequency, such as `student`, `tried`, and `used`. Each processed expert diagnosis is represented as a bag-of-words vector, where an element of the vector indicates the number of occurrences of a term from a global vocabulary. Where we had multiple expert labels available for a single incorrect student answer, we concatenated the two labels and constructed a bag-of-words representation of the result.

		Crossfold Type		
		Evaluator	Prob Type	Seed
Training	Folds	19	14	64
	Data Points	302	296	314
	Evaluators	18	19	19
	Exercises	4	4	4
	Prob Types	14	13	14
	Seeds	61	59	63
Test	Data Points	17	24	5
	Evaluators	1	3	1
	Exercises	2	1	1
	Prob Types	2	1	1
	Seeds	3	5	1

Table 3: Statistics for different cross validation schemes. Entries are rounded averages across folds.

4.4 Mapping Answer Vectors to Diagnoses

With both embeddings of student responses and expert-generated diagnoses in hand, we could explore the extent to which the continuous vector representation of an incorrect answer is related to a semantic description of the misconception underlying that answer. We trained a multinomial logistic regression model to calibrate this correspondence that uses a vector embedding of an incorrect student answer to predict the words in the expert’s diagnosis of that answer. The regression takes as input an m -vector representing a student answer, where m is the dimensionality of the skip-gram embedding space (a hyperparameter of the model). The model produces as output an n -vector, where n is the size of the teacher misconception diagnosis vocabulary. Because of the regression’s use of softmax, this n -vector forms a probability distribution across all terms used in the teacher diagnoses. The i^{th} element of the vector expresses the predicted probability that the i^{th} term of the diagnosis vocabulary applies to the student answer.

5. RESULTS

Here, we describe our results and methodology for evaluating the representations produced by a skip-gram model by using logistic regression and the expert-generated misconception diagnoses. We performed a search over the hyperparameters of the skip-gram algorithm and then compared the predictions generated by our machine learning pipeline to two baselines.

5.1 Skip-Gram Model Evaluation

Recall from Figure 1 that we use logistic regression to train a model identifying correlations between embeddings of student answers and semantic explanations of the underlying misconceptions responsible for incorrect answers. The model surfaces correlations by taking a vector representation as input and producing a probability distribution over the vocabulary of terms used by educators in their misconception diagnoses as output.

Using the semantic data collected from educators as ground truth, we evaluated the insights generated through logistic regression when using vectors produced by different skip-gram models as input. We performed a standard leave-one-out cross-validation (CV) procedure on the educator data. We then evaluated the quality of a model’s predicted misconception tags for student answers in the remaining fold using recall at N , where the value of N for each prediction

is equal to the number of terms used in the original expert label for the relevant incorrect answer. This is defined as:

$$R = \frac{|\hat{T}_N \cap T|}{|T|} \quad (3)$$

where T is the set of terms contained in an educator’s misconception diagnosis for an answer, \hat{T}_N is the set of terms corresponding to the N largest entries in the probability distribution produced by the logistic regression when given an embedding of the answer as input, and $N = |T|$.

We performed three leave-one-out cross-validations using each of the following to determine the fold segmentation:

1. *Evaluator*: The ID of the educator who produced the misconception diagnosis.
2. *Problem Type*: The ID of the template used to generate a question.
3. *Seed*: The unique identifier of an instantiated question.

Descriptive statistics concerning the train and test splits for each scheme are summarized in Table 3.

5.2 Results of Hyperparameter Search

We trained over 750 skip-gram models using different combinations of hyperparameters and then ran each model through the cross-validation procedure described above. The hyperparameters we varied were:

1. *Vector Size*: The number of elements in the vector representations learned by the skip-gram model
2. *Window Size*: The width of each token’s context, i.e., the number of surrounding tokens to consider in the loss function defined in Equation 1.
3. *Min Count*: The minimum number of times a token occurs in the training set to be included in the model.
4. *Training Epochs*

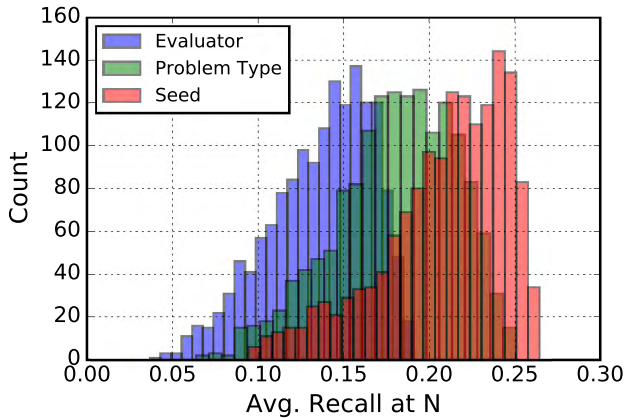


Figure 4: Distribution of average recall under the different cross-validation schemes.

Figure 4 shows the distribution of recall results achieved by all the models under each scheme. We also examined the distribution of hyperparameters among the ten models that achieved the highest average recall at N under each cross-validation type. We found that this metric was not sensitive to the hyperparameter values among the top ten models for all CV types. Within each CV type, all models produced scores within $0.0x$ of one another. Table 4 shows the

hyperparameters that produced the best performing models, measured by average recall, for all CV types.

	Evaluator	Problem Type	Seed
Vector Size	60	100	100
Window Size	15	40	8
Min Count	10	15	5
Training Epochs	20	20	20

Table 4: The best skip-gram hyperparameter combinations under each cross validation scheme.

5.3 Diagnosis Generalization by Best Models

We compared the recall achieved by predicting the words in the diagnoses using the best skip-gram embeddings and logistic regression to the recall achieved by two baseline prediction schemes. For each incorrect student answer, all of the methods predict N terms, where N is the number of terms contained in the original expert diagnosis of the underlying misconception for that answer. This ensures we can fairly measure each prediction scheme by recall at N . The two baselines were:

1. *Random*: Generate a random sample of N terms from the vocabulary formed by the expert misconception diagnoses in the training set.
2. *Frequency*: Predict the N terms that appear most frequently in the diagnoses from the training set.

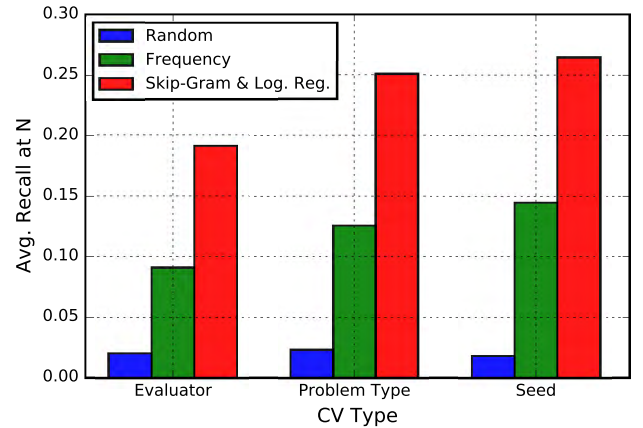


Figure 5: Average recall achieved by different prediction schemes for each cross-validation type.

The average recall at N achieved by the predictions generated through each baseline scheme, as well as that of our own approach, is shown in Figure 5. As expected, a frequency-based approach outperforms a random approach in all three cross-validation types. In addition, the embedding-based approach significantly outperforms the frequency-based approach in all three cases by nearly 100%. The results show that between 18% and 27% of words in held-out diagnoses were recovered. This improvement over baseline suggests a moderate correspondence between the regularities learned in the embedding and semantics used to describe misconceptions.

Recall increased with the size of the training set, with *Seed* having the largest training set and *Evaluator* having the smallest. Other factors may also contribute to these results. First, we chose Khan Academy exercises spanning a diverse selection of mathematical concepts, and the diagnoses for

misconceptions that arise in one domain (e.g., fractions) may use very different diagnosis terms than the terms used for misconceptions in another domain (e.g., surface area). Therefore, there are likely cases where the training set doesn't contain the proper terms to express the misconception diagnoses in the test set. Moreover, different educators used different taxonomies and terms when constructing their misconception diagnoses, which means a model may not be able to accurately predict the diagnoses provided by an educator that isn't well represented in the training data set, which appears to be the situation that arises in *Evaluator* cross-validation.

6. DISCUSSION

Should the 27% recall that we achieved in predicting the terms of held out misconception diagnoses be considered a good score? There are not prior results in this particular area with which to compare to a state of the art. However, this technique of linearly translating from one space (answer embedding) to another (diagnosis bag-of-words) is akin to machine translation from one language's embedding to another. Looking at the accuracy reported in the original linear machine translation paper [9], a translation accuracy of 10% was achieved between English and Vietnamese and 24% translated the other way. Therefore, we could consider 27% a comparable score to past NLP translation benchmarks and a performance level that may produce diagnoses that expert teachers could consider and potentially act on.

A limitation of our approach was that, as discussed in Section 4.2, our survey allowed experts to write open-ended misconception diagnoses which resulted in low frequency of some words and thus a more challenging downstream prediction task. A future study could restrict the terms available for use in expert labels or have them simultaneously negotiate a shared taxonomy. Finally, the student response sequences used as input for the skip-gram models were partitioned by Khan Academy exercise due to us wanting to focus on a limited number of topic areas. This may have lead to missing misconception signatures that manifest or generalize across exercises.

7. ACKNOWLEDGEMENTS

We thank Khan Academy for sharing anonymized exercise data. This work was supported, in part, by a grant from the National Science Foundation (#1547055).

8. REFERENCES

- [1] R. Almond, I. Goldin, Y. Guo, and N. Wang. Vertical and stationary scales for progress maps. In *EDM*, 2014.
- [2] J. R. Anderson. *Rules of the mind*. Psychology Press, 2014.
- [3] J. S. Brown and K. VanLehn. Repair theory: A generative theory of bugs in procedural skills. *Cognitive science*, 4(4):379–426, 1980.
- [4] M. Eagle and T. Barnes. Exploring differences in problem solving with data-driven approach maps. In *Educational Data Mining*, 2014.
- [5] M. Q. Feldman, J. Y. Cho, M. Ong, S. Gulwani, Z. Popović, and E. Andersen. Automatic diagnosis of students' misconceptions in k-8 mathematics. In *CHI '18*, page 264. ACM, 2018.
- [6] R. Liu, R. Patel, and K. R. Koedinger. Modeling common misconceptions in learning process data. In *Proceedings of the Sixth Intl. Conf. on Learning Analytics & Knowledge*, pages 369–377. ACM, 2016.
- [7] T. S. McTavish and J. A. Larusson. Labeling mathematical errors to reveal cognitive states. In *European Conference on Technology Enhanced Learning*, pages 446–451. Springer, 2014.
- [8] J. J. Michalenko, A. S. Lan, and R. G. Baraniuk. Data-mining textual responses to uncover misconception patterns. In *Proceedings of the 10th Conference on Educational Data Mining*, pages 208–213, 2017.
- [9] T. Mikolov, Q. V. Le, and I. Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.
- [10] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
- [11] A. Muehling. Concept landscapes-a new way of using concept maps. *Journal of Educational Data Mining*, 9(2):1–30, 2017.
- [12] Z. A. Pardos and A. Dadu. Imputing kcs with representations of problem content and context. In *UMAP '17*, pages 148–155. ACM, 2017.
- [13] Z. A. Pardos, Z. Fan, and W. Jiang. Connectionist recommendation in the wild: On the utility and scrutability of neural networks for personalized course guidance. *User Modeling and User-Adapted Interaction*, 2019.
- [14] Z. A. Pardos, S. Farrar, J. Kolb, G. X. Peh, and J. H. Lee. Distributed Representation of Misconceptions. In J. Kay and R. Luckin, editors, *Proceedings of the 13th International Conference of the Learning Sciences (ICLS)*, pages 1791–1798, London, UK, 2018.
- [15] R. Pelánek and J. Rihák. Properties and applications of wrong answers in online educational systems. In *EDM*, pages 466–471, 2016.
- [16] J. Piaget. The child's concept of number, 1952.
- [17] A. Schoenfeld. Personal Communication.
- [18] D. Selent and N. Heffernan. Reducing student hint use by creating buggy messages from machine learned incorrect processes. In *Intl. Conf. on Intelligent Tutoring Systems*, pages 674–675. Springer, 2014.
- [19] J. P. Smith, A. A. DiSessa, and J. Roschelle. Misconceptions reconceived: a constructivist analysis of knowledge in transition. *The journal of the learning sciences*, 3(2):115–163, 1994.
- [20] W. L. J.-E. Soloway. Intention-based diagnosis of programming errors. In *Proceedings of the 5th National Conference on Artificial Intelligence, Austin, TX*, pages 162–168, 1984.
- [21] M. Straatemeier et al. Math garden: A new educational and scientific instrument. *Education*, 57:1813–1824, 2014.
- [22] K. K. Tatsuoaka. A probabilistic model for diagnosing misconceptions by the pattern classification approach. *Journal of Educational Statistics*, 10(1):55–73, 1985.
- [23] L. S. Vygotsky. *Mind in society: The development of higher psychological processes*. Harvard university press, 1980. Original Manuscripts ca. 1930–1934.

Active Learning of Strict Partial Orders: A Case Study on Concept Prerequisite Relations

Chen Liang¹ Jianbo Ye¹ Han Zhao² Bart Pursel¹ C. Lee Giles¹

¹The Pennsylvania State University

²Carnegie Mellon University

{cul226, jxy198, bkp10, clg20}@psu.edu

han.zhao@cs.cmu.edu

ABSTRACT

Strict partial order is a mathematical structure commonly seen in relational data. One obstacle to extracting such type of relations at scale is the lack of large scale labels for building effective data-driven solutions. We develop an active learning framework for mining such relations subject to a strict order. Our approach incorporates relational reasoning not only in finding new unlabeled pairs whose labels can be deduced from an existing label set, but also in devising new query strategies that consider the relational structure of labels. Our experiments on concept prerequisite relations show our proposed framework can substantially improve the classification performance with the same query budget compared to other baseline approaches.

1. INTRODUCTION

Pool-based active learning is a learning framework where the learning algorithm is allowed to access a set of unlabeled examples and ask for the labels of any of these examples [3]. Its goal is to learn a good classifier with significantly fewer labels by actively directing the queries to the most “valuable” examples. In a typical setup of active learning, the label dependency among labeled or unlabeled examples is not considered. But data and knowledge in the real world are often embodied with prior relational structures. Taking into consideration those structures in building machine learning solutions can be necessary and crucial. The goal of this paper is to investigate the query strategies in active learning of a strict partial order, namely, when the ground-truth labels of examples constitute an irreflexive and transitive relation. In this paper, we develop efficient and effective algorithms extending popular query strategies used in active learning to work with such relational data. We study the following problem in the active learning context:

Problem. Given a finite set V , a strict order on V is a type of irreflexive and transitive (pairwise) relation. Such a strict order is represented by a subset $G \subseteq V \times V$. Given an unknown strict order G , an oracle W that returns $W(u, v) =$

$-1 + 2 \cdot \mathbf{1}[(u, v) \in G] \in \{-1, 1\}$, and a feature extractor $\mathcal{F} : V \times V \mapsto \mathbb{R}^d$, find $h : \mathbb{R}^d \mapsto \{-1, 1\}$ from a hypothesis class \mathcal{H} that predicts whether or not $(u, v) \in G$ for each pair $(u, v) \in V \times V$ and $u \neq v$ (using $h(\mathcal{F}(u, v))$) by querying W a finite number of (u, v) pairs from $V \times V$.

Our main focus is to develop reasonable query strategies in active learning of a strict order exploiting both the knowledge from (non-consistent) classifiers trained on a limited number of labeled examples and the deductive structures among pairwise relations. Our work also has a particular focus on *partial orders*. If the strict order is total, a large school called “learning to rank” has studied this topic [10], some of which are under the active learning setting [4]. Learning to rank relies on binary classifiers or probabilistic models that are consistent with the rule of a total order. Such approaches are however limited in a sense to principally modeling a partial order: a classifier consistent with a total order will always have a non-zero lower bound of error rate, if the ground-truth is a partial order but not a total order.

In our active learning problem, incorporating the deductive relations of a strict order in soliciting examples to be labeled is non-trivial and important. The challenges motivating us to pursue this direction can be explained in three folds: First, any example whose label can be deterministically reasoned from a labeled set by using the properties of strict orders does not need further manual labeling or statistical prediction. Second, probabilistic inference of labels based on the independence hypothesis, as is done in the conventional classifier training, is *not* proper any more because the deductive relations make the labels of examples dependent on each other. Third, in order to quantify how valuable an example is for querying, one has to combine uncertainty and logic to build proper representations. Sound and efficient heuristics with empirical success are to be explored.

One related active learning work that deals with a similar setting to ours is [13], whereas equivalence relations are considered instead. Particularly, they made several crude approximations in order to expedite the expected error calculation to a computational tractable level. We approach the design of query strategies from a different perspective while keeping efficiency as one of our central concerns.

To empirically study the proposed active learning algorithm, we apply it to *concept prerequisite learning problem* [15, 8], where the goal is to predict whether a concept A is a pre-

Chen Liang, Jianbo Ye, Han Zhao, Bart Pursel and C. Lee Giles
"Active Learning of Strict Partial Orders: A Case Study on
Concept Prerequisite Relations" In: *Proceedings of The 12th
International Conference on Educational Data Mining (EDM
2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, &
Roger Nkambou (eds.) 2019, pp. 348 - 353

requisite of a concept B given the pair (A, B) . Although there have been some research efforts towards learning prerequisites [16, 15, 8, 17], the mathematical nature of the prerequisite relation as strict partial orders has not been investigated. In addition, one obstacle for effective learning-based solutions to this problem is the lack of large scale prerequisite labels. Liang et al. [9] applied standard active learning to this problem without utilizing relation properties of prerequisites. Active learning methods tailored for strict partial orders provide a good opportunity to tackle the current challenges of concept prerequisite learning.

Our main contributions are summarized as follows: First, we propose a new efficient reasoning module for monotonically calculating the deductive closure under the assumption of a strict order. This computational module can be useful for general AI solutions that need fast reasoning in regard to strict orders. Second, we apply our reasoning module to extend two popular active learning approaches to handle relational data and empirically achieve substantial improvements. This is the first attempt to design active learning query strategies tailored for strict partial orders. Third, under the proposed framework, we solve the problem of concept prerequisite learning and our approach appears to be successful on data from four educational domains, whereas previous work have not exploited the relational structure of prerequisites as strict partial orders in a principled way.

2. REASONING OF A STRICT ORDER

2.1 Preliminary

DEFINITION 1 (STRICT ORDER). Given a finite set V , a subset G of $V \times V$ is called a strict order if and only if it satisfies the two conditions: (i) if $(a, b) \in G$ and $(b, c) \in G$, then $(a, c) \in G$; (ii) if $(a, b) \in G$, then $(b, a) \notin G$.

DEFINITION 2 (G-ORACLE). For two subsets $G, H \subseteq V \times V$, a function denoted as $W_H(\cdot, \cdot) : H \mapsto \{-1, 1\}$ is called a G -oracle on H iff for any $(u, v) \in H$, $W_H(u, v) = -1 + 2 \cdot \mathbf{1}[(u, v) \in G]$.

The G -oracle returns a label denoting whether a pair belongs to G .

DEFINITION 3 (COMPLETENESS OF AN ORACLE). A G -oracle of strict order W_H is called complete if and only if H satisfies: for any $a, b, c \in V$, (i) if $(a, b) \in H \cap G$, $(b, c) \in H \cap G$, then $(a, c) \in H \cap G$; (ii) if $(a, b) \in H \cap G$, $(a, c) \in H \cap G^c$, then $(b, c) \in H \cap G^c$; (iii) if $(b, c) \in H \cap G$, $(a, c) \in H \cap G^c$, then $(a, b) \in H \cap G^c$; (iv) if $(a, b) \in H \cap G$, then $(b, a) \in H \cap G^c$, where G^c is the complement of G . W_H is called complete if it is consistent under transitivity when restricted on pairs from H .

DEFINITION 4 (CLOSURE). Given a strict order G , for any $H \subseteq V \times V$, its closure is defined to be the smallest set \bar{H} such that $H \subseteq \bar{H}$ and the G -oracle $W_{\bar{H}}$ is complete.

DEFINITION 5 (DESCENDANT AND ANCESTOR). Given a strict order G of V and $a \in V$, its ancestor subject to G is $A_a^G := \{b \mid (b, a) \in G\}$ and its descendant is $D_a^G := \{b \mid (a, b) \in G\}$.

2.2 Reasoning Module for Closure Calculation

With the definitions in the previous section, this section proposes a reasoning module that is designed to monotonically calculate the deductive closure for strict orders. Remark

that a key difference between the traditional transitive closure and our definition of closure (Definition 3&4) is that the former only focuses on G but the latter requires calculation for both G and G^c . In the context of machine learning, relations in G and G^c correspond to positive examples and negative examples, respectively. Since both of these examples are crucial for training classifiers, existing algorithms for calculating transitive closure such as the Warshall algorithm are not applicable. Thus we propose the following theorem for monotonically computing the closure. Please refer to supplemental material for the proofs.

THEOREM 1. Let G be a strict order of V and W_H a complete G -oracle on $H \subseteq V \times V$. For any pair $(a, b) \in V \times V$, define the notation $C_{(a,b)}$ by

(i) If $(a, b) \in H$, $C_{(a,b)} := H$.

(ii) If $(a, b) \in G^c \cap H^c$, $C_{(a,b)} := H \cup N'_{(a,b)}$ where

$$N'_{(a,b)} := \{(d, c) \mid c \in A_a^{G \cap H} \cup \{b\}, d \in D_b^{G \cap H} \cup \{a\}\},$$

and particularly $N'_{(a,b)} \subseteq G^c$.

(iii) If $(a, b) \in G \cap H^c$, $C_{(a,b)} := H \cup N_{(a,b)} \cup R_{(a,b)} \cup S_{(a,b)} \cup T_{(a,b)} \cup O_{(a,b)}$, where

$$N_{(a,b)} := \{(c, d) \mid c \in A_a^{G \cap H} \cup \{a\}, d \in D_b^{G \cap H} \cup \{b\}\},$$

$$R_{(a,b)} := \{(d, c) \mid (c, d) \in N_{(a,b)}\},$$

$$S_{(a,b)} := \{(d, e) \mid c \in A_a^{G \cap H} \cup \{a\}, d \in D_b^{G \cap H} \cup \{b\}, (c, e) \in G^c \cap H\},$$

$$T_{(a,b)} := \{(e, c) \mid c \in A_a^{G \cap H} \cup \{a\}, d \in D_b^{G \cap H} \cup \{b\}, (e, d) \in G^c \cap H\},$$

$$O_{(a,b)} := \bigcup_{(c,d) \in S_{(a,b)} \cup T_{(a,b)}} N''_{(c,d)},$$

$$N''_{(c,d)} := \{(f, e) \mid e \in A_d^{G \cap (H \cup N_{(a,b)})} \cup \{d\}, f \in D_c^{G \cap (H \cup N_{(a,b)})} \cup \{c\}\}.$$

In particular, $N_{(a,b)} \subseteq G$ and $R_{(a,b)} \cup S_{(a,b)} \cup T_{(a,b)} \cup O_{(a,b)} \subseteq G^c$.

For any pair $(x, y) \in V \times V$, the closure of $H' = H \cup \{(x, y)\}$ is $C_{(x,y)}$.

Figure 1 provides an informal explanation of each necessary condition (except for $R_{(a,b)}$) mentioned in the theorem. If (a, b) is a positive example, i.e. $(a, b) \in G$, then (i) $N_{(a,b)}$ is a set of inferred positive examples by transitivity; (ii) $R_{(a,b)}$ is a set of inferred negative examples by irreflexivity; (iii) $S_{(a,b)}$ and $T_{(a,b)}$ are sets of inferred negative examples by transitivity; (iv) $O_{(a,b)}$ is a set of negative examples inferred from $S_{(a,b)}$ and $T_{(a,b)}$. If (a, b) is a negative example, i.e. $(a, b) \in G^c$, then $N'_{(a,b)}$ is a set of negative examples inferred by transitivity.

3. POOL-BASED ACTIVE LEARNING

The pool-based sampling [7] is a typical active learning scenario in which one maintains a labeled set \mathcal{D}_l and an unlabeled set \mathcal{D}_u . In particular, we let $\mathcal{D}_u \cup \mathcal{D}_l = \mathcal{D} = \{1, \dots, n\}$ and $\mathcal{D}_u \cap \mathcal{D}_l = \emptyset$. For $i \in \{1, \dots, n\}$, we use $\mathbf{x}_i \in \mathbb{R}^d$ to denote a feature vector representing the i -th instance, and $y_i \in \{-1, +1\}$ to denote its groundtruth class label. At each round, one or more instances are selected from \mathcal{D}_u whose label(s) are then requested, and the labeled instance(s) are

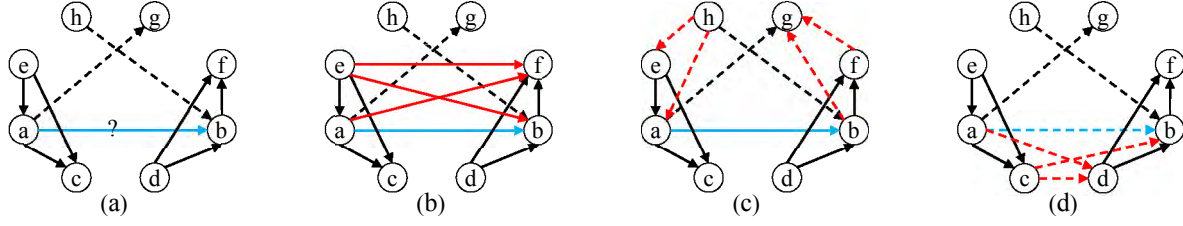


Figure 1: Following the notations in Theorem 1: (a) Black lines are pairs in H , solid lines are pairs in G , and dashed lines are pairs in G^c . The pair (a, b) in the cyan color is the pair to be labeled or deduced. (b) If $(a, b) \in G$, $\{(a, b), (e, f), (a, f), (e, b)\} \subseteq N_{(a, b)}$. (c) If $(a, b) \in G$, $\{(h, e), (h, a)\} \subseteq T_{(a, b)}$ and $\{(b, g), (f, g)\} \subseteq S_{(a, b)}$. (d) If $(a, b) \in G^c$, $\{(a, b), (a, d), (c, b), (c, d)\} \subseteq N'_{(a, b)}$. Likewise, if $\exists (x, y) \in G$, s.t. $(a, b) \in S_{(x, y)} \cup T_{(x, y)}$, $\{(a, b), (a, d), (c, b)\} \subseteq O_{(x, y)}$.

then moved to \mathcal{D}_l . Typically instances are queried in a prioritized way such that one can obtain good classifiers trained with a substantially smaller set \mathcal{D}_l . We focus on the pool-based sampling setting where queries are selected in serial, i.e., one at a time.

3.1 Query Strategies

The key component of active learning is the design of an effective criterion for selecting the most “valuable” instance to query, which is often referred to as *query strategy*. We use s^* to refer to the selected instance by the strategy. In general, different strategies follow a greedy framework:

$$s^* = \underset{s \in \mathcal{D}_u}{\operatorname{argmax}} \min_{y \in \{-1, 1\}} f(s; y, \mathcal{D}_l), \quad (1)$$

where $f(s; y, \mathcal{D}_l) \in \mathbb{R}$ is a scoring function to measure the risks of choosing y as the label for $\mathbf{x}_s \in \mathcal{D}_u$ given an existing labeled set \mathcal{D}_l .

We investigate two commonly used query strategies: uncertainty sampling [6] and query-by-committee [14]. We show that under the binary classification setting, they can all be reformulated as Eq. (1).

Uncertainty Sampling selects the instance which it is least certain how to label. We choose to study one popular uncertainty-based sampling variant, the *least confident*. Subject to Eq. (1), the resulting approach is to let

$$f(s; y, \mathcal{D}_l) = 1 - P_{\Delta(\mathcal{D}_l)}(y_s = y | \mathbf{x}_s), \quad (2)$$

where $P_{\Delta(\mathcal{D}_l)}(y_s = y | \mathbf{x}_s)$ is a conditional probability which is estimated from a probabilistic classification model Δ trained on $\{(\mathbf{x}_i, y_i) \mid \forall i \in \mathcal{D}_l\}$.

Query-By-Committee maintains a committee of models trained on labeled data, $\mathcal{C}(\mathcal{D}_l) = \{g^{(1)}, \dots, g^{(C)}\}$. It aims to reduce the size of version space. Specifically, it selects the unlabeled instance about which committee members disagree the most based on their predictions. Subject to Eq. (1), the resulting approach is to let

$$f(s; y, \mathcal{D}_l) = \sum_{k=1}^C \mathbf{1}[y \neq g^{(k)}(\mathbf{x}_s)], \quad (3)$$

where $g^{(k)}(\mathbf{x}_s) \in \{-1, 1\}$ is the predicted label of \mathbf{x}_s using the classifier $g^{(k)}$.

Our paper will start from generalizing Eq. (1) and show that it is possible to extend the two popular query strategies for considering relational data as a strict order.

4. ACTIVE LEARNING OF A STRICT ORDER

Given G a strict order of V , consider a set of data $\mathcal{D} \subseteq V \times V$, where $(a, a) \notin \mathcal{D}, \forall a \in V$. Similar to the pool-based active learning, one needs to maintain a labeled set \mathcal{D}_l and an unlabeled set \mathcal{D}_u . We require that $\mathcal{D} \subseteq \mathcal{D}_l \cup \mathcal{D}_u$ and $\mathcal{D}_l \cap \mathcal{D}_u = \emptyset$. Given a feature extractor $\mathcal{F}: V \times V \mapsto \mathbb{R}^d$, we can build a vector dataset $\{\mathbf{x}_{(a, b)} = \mathcal{F}(a, b) \in \mathbb{R}^d \mid (a, b) \in \mathcal{D}\}$. Let $y_{(a, b)} = -1 + 2 \cdot \mathbf{1}[(a, b) \in G] \in \{-1, 1\}$ be the ground-truth label for each $(a, b) \in V \times V$. Active learning aims to query Q a subset from \mathcal{D} under limited budget and construct a label set \mathcal{D}_l from Q , in order to train a good classifier h on $\mathcal{D}_l \cap \mathcal{D}$ such that it predicts accurately whether or not an unlabeled pair $(a, b) \in G$ by $h(\mathcal{F}(a, b)) \in \{-1, 1\}$.

Active learning of strict orders differs from the traditional active learning in two unique aspects: (i) By querying the label of a single unlabeled instance, one may obtain a set of labeled examples, with the help of strict orders’ properties; (ii) The relational information of strict orders could also be utilized by query strategies. We will present our efforts towards incorporating the above two aspects into active learning of a strict order.

4.1 Basic Relational Reasoning in Active Learning

A basic extension from standard active learning to one under the strict order setting is to apply relational reasoning when both updating \mathcal{D}_l and predicting labels. Algorithm 1 shows the pseudocode for the pool-based active learning of a strict order. When updating \mathcal{D}_l with a new instance $(a, b) \in \mathcal{D}_u$ whose label $y_{(a, b)}$ is acquired from querying, one first calculates $\overline{\mathcal{D}_l}$, i.e., the closure of $\mathcal{D}_l \cup \{(a, b)\}$, using Theorem 1, and then sets $\mathcal{D}_l := \overline{\mathcal{D}_l}$ and $\mathcal{D}_u := \mathcal{D} \setminus \overline{\mathcal{D}_l}$ respectively. Therefore, it is possible to augment the labeled set \mathcal{D}_l with more than one pair at each stage even though only a single instance is queried. Furthermore, the following corollary shows that given a fixed set of samples to be queried, their querying order does not affect the final labeled set \mathcal{D}_l constructed.

COROLLARY 1.1. *Given a list of pairs Q of size m whose elements are from $V \times V$, let i_1, \dots, i_m and j_1, \dots, j_m be two different permutations of $1, \dots, m$. Let $I_0 = \emptyset$ and $J_0 = \emptyset$, and $I_k = I_{k-1} \cup \{q_{i_k}\}$, $J_k = J_{k-1} \cup \{q_{j_k}\}$ for $k = 1, \dots, m$, where $\bar{\cdot}$ is defined as the closure set under G . We have $I_m = J_m$, which is the closure of $\{q_i \in V \times V \mid i = 1, \dots, m\}$.*

Corollary 1.1 is a straightforward result from the uniqueness of closure, which is also verified by our experiments. The la-

Algorithm 1 Pseudocode for pool-based active learning of a strict order.

Input:
 $\mathcal{D} \subseteq V \times V$ % a data set

Initialize:
 $\mathcal{D}_l \leftarrow \{(a_{s_1}, b_{s_1}), (a_{s_2}, b_{s_2}), \dots, (a_{s_k}, b_{s_k})\}$ % initial labeled set with k seeds
 $\mathcal{D}_l \leftarrow \overline{\mathcal{D}_l}$ % initial closure
 $\mathcal{D}_u \leftarrow \mathcal{D} \setminus \mathcal{D}_l$ % initial unlabeled set

while $\mathcal{D}_u \neq \emptyset$ **do**
 Select (a^*, b^*) from \mathcal{D}_u % according to a query strategy
 Query the label $y_{(a^*, b^*)}$ for the selected instance (a^*, b^*)
 $\mathcal{D}_l \leftarrow \mathcal{D}_l \cup \{(a^*, b^*)\}$
 $\mathcal{D}_u \leftarrow \mathcal{D} \setminus \mathcal{D}_l$
end while

beled set \mathcal{D}_l contains two kinds of pairs based on where their labels come from: The first kind of labels comes directly from queries, and the second kind comes from the relational reasoning as explained by Theorem 1. Such an approach has a clear advantage over standard active learning at the same budget of queries, because labels of part of the test pairs can be inferred deterministically and as a result there will be more labeled data for supervised training. In our setup of active learning, we train classifiers on $\mathcal{D} \cap \mathcal{D}_l$ and use them for predicting the labels of remaining pairs that are not in \mathcal{D}_l .

4.2 Query Strategies with Relational Reasoning

The relational active learning framework as explained in the previous section however does not consider incorporating relational reasoning in its query strategy. We further develop a systematic approach on how to achieve this.

We start from the following formulation: at each stage, one chooses a pair (a^*, b^*) to query based on

$$(a^*, b^*) = \operatorname{argmax}_{(a,b) \in \mathcal{D}_u} \min_{y \in \{-1, 1\}} F(\mathcal{S}(y_{(a,b)} = y), \mathcal{D}_l), \quad (4)$$

$$\mathcal{S}(y_{(a,b)} = y) = (\overline{\mathcal{D}_l \cup \{(a,b)\}} \setminus \mathcal{D}_l) \cap \mathcal{D}. \quad (5)$$

Again, F is the scoring function. $\mathcal{S}(y_{(a,b)} = y)$ is the set of pairs in \mathcal{D} whose labels, originally unknown ($\notin \mathcal{D}_l$), can now be inferred by assuming $y_{(a,b)} = y$ using Theorem 1. For each $(u, v) \in \mathcal{S}(y_{(a,b)} = y)$, its inferred label is denoted as $\hat{y}_{(u,v)}$ in the sequel. One can see that this formulation is a generalization of Eq. (1). We now proceed to develop extensions for the two query strategies to model the dependencies between pairs imposed by the rule of a strict order. Following the same notations as previously described with the only difference that the numbering index is replaced by the pairwise index, we propose two query strategies tailored to strict orders.

Uncertainty Sampling with Reasoning. With relational reasoning, one not only can reduce the uncertainty of the queried pair (a, b) but also may reduce that of other pairs deduced

Table 1: Dataset statistics.

Domain	# Concepts	# Pairs	# Prerequisites
Data Mining	120	826	292
Geometry	89	1681	524
Physics	153	1962	487
Precalculus	224	2060	699

by assuming $y_{(a,b)} = y$. The modified scoring function reads:

$$F(\mathcal{S}(y_{(a,b)} = y), \mathcal{D}_l) = \sum_{(u,v) \in \mathcal{S}(y_{(a,b)} = y)} 1 - P_{\Delta(\mathcal{D}_l \cap \mathcal{D})}(y_{(u,v)} = \hat{y}_{(u,v)} | \mathbf{x}_{(u,v)}). \quad (6)$$

Query-by-Committee with Reasoning. Likewise, one also has the extension for QBC, where $\{g^{(k)}\}_{k=1}^C$ is a committee of classifiers trained on bagging samples of $\mathcal{D}_l \cap \mathcal{D}$,

$$F(\mathcal{S}(y_{(a,b)} = y), \mathcal{D}_l) = \sum_{(u,v) \in \mathcal{S}(y_{(a,b)} = y)} \sum_{k=1}^C \mathbf{1}(\hat{y}_{(u,v)} \neq g^{(k)}(\mathbf{x}_{(u,v)})). \quad (7)$$

5. EXPERIMENTS

For evaluation, we apply the proposed active learning algorithms to *concept prerequisite learning problem* [8]. Given a pair of concepts (A, B) , we predict whether or not A is a prerequisite of B , which is a binary classification problem. Here, cases where B is a prerequisite of A and where no prerequisite relation exists are both considered negative.

5.1 Dataset

We use the Wiki concept map dataset from [17] which is collected from textbooks on different educational domains. Each concept corresponds to an English Wiki article. For each domain, the dataset consists of prerequisite pairs in the concept map. Table 1 summarizes the statistics of the our final processed dataset.

5.2 Features

For each concept pair (A, B) , we calculate two types of features following the popular practice of information retrieval and natural language processing: graph-based features and text-based features. Please refer to Table 2 for detailed description. Note we trained a topic model [1] on the Wiki corpus. We also trained a Word2Vec [12] model on the same corpus with each concept treated as an individual token.

5.3 Experiment Settings

We follow the typical evaluation protocol of pool-based active learning. We first randomly split a dataset into a training set \mathcal{D} and a test set \mathcal{D}_{test} with a ratio of 2:1. Then we randomly select 20 samples from the training set as the initial query set Q and compute its closure \mathcal{D}_l . Meanwhile, we set $\mathcal{D}_u = \mathcal{D} \setminus \mathcal{D}_l$. In each iteration, we pick an unlabeled instance from \mathcal{D}_u to query for its label, update the label set \mathcal{D}_l , and re-train a classification model on the updated $\mathcal{D}_l \cap \mathcal{D}$. The re-trained classification model is then evaluated on \mathcal{D}_{test} . In all experiments, we use a random forests classifier [2] with 200 trees as the classification model. We use Area under the ROC curve (AUC) as the evaluation metric. Taking into account the effects of randomness subject to different initializations, we continue the above experimental process for each method repeatedly with 300 pre-selected distinct random seeds. Their average scores and

Table 2: Feature description. Top: graph-based features. Bottom: text-based features.

Feature	Description
In/Out Degree	The in/out degree of A/B.
Common Neighbors	# common neighbors of A and B.
# Links	# times A/B links to B/A.
Link Proportion	The proportion of pages that link to A/B also link to B/A.
NGD	The Normalized Google Distance between A and B [18].
PMI	The Pointwise Mutual Information relatedness between the incoming links of A and B.
RefD	A metric to measure how differently A and B's related concepts refer to each other [8].
HITS	The difference between A and B's hub/authority scores. [5]
1st Sent	Whether A/B is in the first sentence of B/A.
In Title	Whether A appears in B's title.
Title Jaccard	The Jaccard similarity between A and B's titles.
Length	# words of A/B's content.
Mention	# times A/B are mentioned in the content of B/A.
NP	# noun phrases in A/B's content; # common noun phrases.
Tf-idf Sim	The cosine similarity between Tf-idf vectors for A and B's first paragraphs.
Word2Vec Sim	The cosine similarity between vectors of A and B trained by Word2Vec.
LDA Entropy	The Shannon entropy of the LDA vector of A/B.
LDA Cross Entropy	The cross entropy between the LDA vector of A/B and B/A.

Table 3: Summary of compared query strategies.

Method	Use reasoning when updating \mathcal{D}_l	Use reasoning to select the instance to query	Use learning to select the instance to query
Random	✗	✗	✗
LC, QBC	✗	✗	✓
Random-R	✓	✗	✗
LC-R, QBC-R	✓	✗	✓
CNT	✓	✓	✗
LC-R+, QBC-R+	✓	✓	✓

confidence intervals ($\alpha = 0.05$) are reported. We compare four query strategies: (i) **Random**: randomly select an instance to query; (ii) **LC**: least confident sampling, a widely used uncertainty sampling variant. We use logistic regression to estimate posterior probabilities; (iii) **QBC**: query-by-committee algorithm. We apply query-by-bagging [11] and use a committee of three decision trees; (iv) **CNT**: a simple baseline query strategy designed to greedily select an instance whose label can potentially infer the most number of unlabeled instances. Following the previous notations, the scoring function for CNT is $F(\mathcal{S}(y_{(a,b)} = y), \mathcal{D}_l) = |S(y_{(a,b)} = y)|$, which is solely based on logical reasoning.

For experiments, we test each query strategy under three settings: (i) Traditional active learning where no relational information is considered. Query strategies under this setting are denoted as Random, LC, and QBC. (ii) Relational active learning where relation reasoning is applied to updating \mathcal{D}_l and predicting labels of \mathcal{D}_{test} . Query strategies under this setting are denoted as Random-R, LC-R, and QBC-R. (iii) Besides being applied to updating \mathcal{D}_l , relational reasoning is also incorporated in the query strategies. Query strategies under this setting are the baseline method CNT and our proposed extensions of LC and QBC for strict partial orders, denoted as LC-R+ and QBC-R+, respectively. Table 3 summarizes the query strategies studied in the experiments.

5.4 Experiment Results

Figure 2 shows the AUC results of different query strategies. For each case, we present the average values and 95% C.I. of repeated 300 trials with different train/test splits. In addition, Figure 3 compares the relations between the number of queries and the number of labeled instances across different query strategies. Note that in the relational active learning setting querying a single unlabeled instance will result in one or more labeled instances. According to Figure 2 and Figure 3, we have the following observations: First, by comparing query strategies under the settings (ii) and (iii) with setting (i), we observe that incorporating relational reasoning into active learning substantially improves the AUC performance of each query strategy. In addition, we find the query order, which is supposed to be different for each strategy, does not affect \mathcal{D}_l at the end when $\mathcal{D} \subseteq \mathcal{D}_l$. Thus, it partly verifies Corollary 1.1. Second, our proposed LC-R+ and QBC-R+ significantly outperform other compared query strategies. Specifically, when comparing them with LC-R and QBC-R, we see that incorporating relational reasoning into directing the queries helps to train a better classifier. Figure 3 shows that LC-R+ and QBC-R+ lead to more labeled instances when using the same amount of queries than that of LC-R and QBC-R. This partly contributes to the performance gain. Third, LC-R+ and QBC-R+ are more effective at both collecting a larger labeled set and training better classifiers than the CNT baseline. In addition, by comparing CNT with LC-R, QBC-R, and Random-R, we observe that a larger size of the labeled set does not always lead to a better performance. Such observations demonstrate the necessity of combining deterministic relational reasoning and probabilistic machine learning in designing query strategies.

In addition to effectiveness, we also conduct empirical studies on the runtime of the reasoning module and include the results in the supplemental material.

6. CONCLUSION

We propose an active learning framework tailored to relational data in the form of strict partial orders. An efficient reasoning module is proposed to extend two commonly used query strategies – uncertainty sampling and query by committee. Experiments on concept prerequisite learning show that incorporating relational reasoning in both selecting valuable examples to label and expanding the training set significantly improves standard active learning approaches. Future work could be to explore the following: (i) apply the reasoning module to extend other query strategies; (ii) active learning of strict partial orders from a noisy oracle.

7. REFERENCES

- [1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
- [2] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [3] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4(1):129–145, 1996.
- [4] P. Donmez and J. G. Carbonell. Optimizing estimated loss reduction for active sampling in rank learning. In *Proc. ICML*, pages 248–255. ACM, 2008.

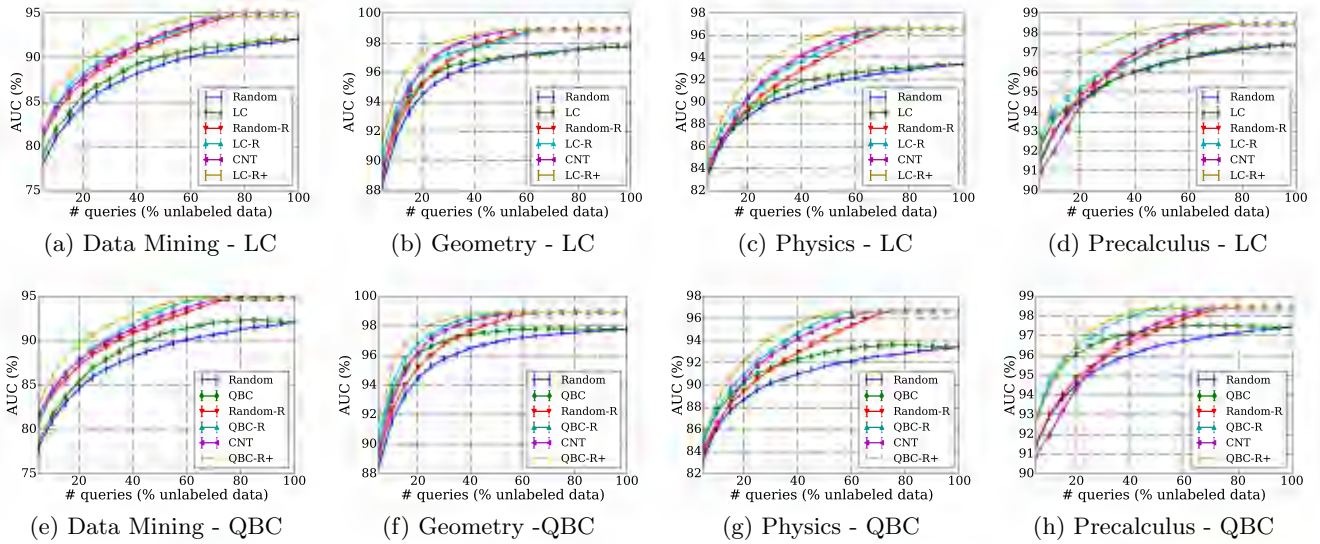


Figure 2: Comparison of different query strategies' AUC scores for concept prerequisite learning.

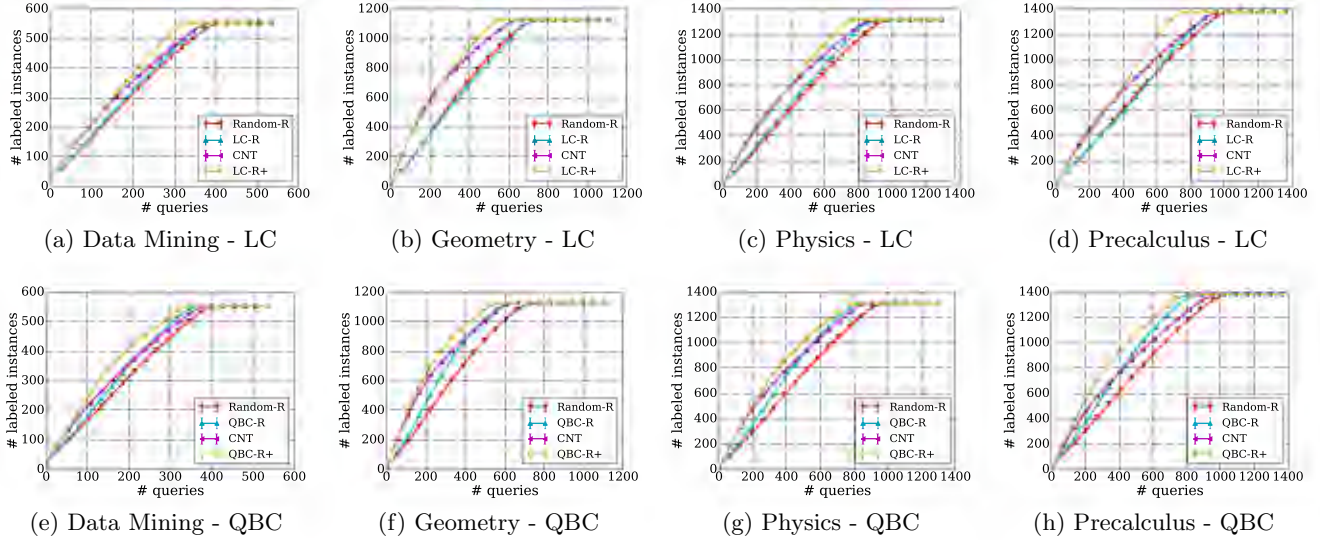


Figure 3: Comparison of relations between the number of queries and the number of labeled instances when using different query strategies.

- [5] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [6] D. D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proc. ICML*, pages 148–156, 1994.
- [7] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proc. SIGIR*, pages 3–12, 1994.
- [8] C. Liang, Z. Wu, W. Huang, and C. L. Giles. Measuring prerequisite relations among concepts. In *Proc. EMNLP*, pages 1668–1674, 2015.
- [9] C. Liang, J. Ye, S. Wang, B. Pursel, and C. L. Giles. Investigating active learning for concept prerequisite learning. In *Proc. EAAI*, 2018.
- [10] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.
- [11] N. A. H. Mamitsuka. Query learning strategies using boosting and bagging. In *Proc. ICML*, volume 1. Morgan Kaufmann Pub, 1998.
- [12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proc. NIPS*, pages 3111–3119, 2013.
- [13] S. Rendle and L. Schmidt-Thieme. Active learning of equivalence relations by minimizing the expected loss using constraint inference. In *Proc. ICDM*, pages 1001–1006. IEEE, 2008.
- [14] H. S. Seung, M. Oppor, and H. Sompolinsky. Query by committee. In *Proc. COLT*, pages 287–294. ACM, 1992.
- [15] P. P. Talukdar and W. W. Cohen. Crowdsourced comprehension: predicting prerequisite structure in Wikipedia. In *Proc. BEA*, pages 307–315. ACL, 2012.
- [16] A. Vuong, T. Nixon, and B. Towle. A method for finding prerequisites within a curriculum. In *Proc. EDM*, pages 211–216, 2011.
- [17] S. Wang, A. Ororbia, Z. Wu, K. Williams, C. Liang, B. Pursel, and C. L. Giles. Using prerequisites to extract concept maps from textbooks. In *Proc. CIKM*, pages 317–326. ACM, 2016.
- [18] I. Witten and D. Milne. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy*, pages 25–30, 2008.

Characterising Students' Writing Processes Using Temporal Keystroke Analysis

Donia Malekian^a
malekiand@unimelb.edu.au

James Bailey^a
baileyj@unimelb.edu.au

Gregor Kennedy^b
gek@unimelb.edu.au

Paula de Barba^b
paula.debarba@unimelb.edu.au

Sadia Nawaz^a
nawazs@student.unimelb.edu.au

^a School of Computing and Information Systems, ^b Melbourne Centre for the Study of Higher Education, University of Melbourne

ABSTRACT

This work aims to characterize students' writing processes using keystroke logs and understand how the extracted characteristics influence the text quality at specific moments of writing. Earlier works have proposed predictive models characterizing students' writing processes and mainly rely on distribution-based measures of pauses obtained from the overall keystroke logs. However, the effect of isolated phases of writing has not been evaluated in these models. Moreover, current theories on writing suggest that the quality of writing depends on when specific writing behaviours are performed. This view is not examined in the keystroke logging analysis literature. Addressing the mentioned challenges, the two contributions of this work are: a) characterizing students' writing processes connected to isolated writing phases and examining their influence on writing quality; and b) temporal analysis of keystrokes and investigating whether the significance of writing characteristics varies as students progress in their writing task. Our results suggest that characterizing students' writing based on isolated writing phases is slightly more predictive of writing quality. Additionally, the effect of several writing characteristics on writing quality changes when considering the time dimension.

Keywords

Writing process, Keystroke log, XGBoost, SHAP feature importance, Temporal analysis

1. INTRODUCTION

The recognized significance of the writing process has led to the emergence of a wide variety of research exploring the process of students' academic writing. The writing process, broadly including planning, writing and revising phases, is a non-linear process [10]. These phases often occur simultaneously, making it challenging for researchers to examine features related to specific writing phases.

One stream of writing research has focused on analyzing pauses in keystroke logs and associating them with different phases of the writing process [9]. This is often accomplished by exploring the distribution of pauses and then mapping the related parameters to specific writing processes. Although some keystroke log features can be a marker of a high writing quality, there is not always sufficient evidence for their relationship. This may be due to decisions made during data processing and analysis.

Additionally, statistics and features collected by most studies examining the relationship between keystroke logs and the writing process mainly rely on aggregated or distribution-based representations of the overall keystroke logs [9]. Even when the data has been summarized to a high standard, neglecting the time dimension may hide the effect of specific behaviors at particular moments of the writing process. Temporal analysis has been suggested as a better way to uncover the writing process and its related stages [3]. However, there are a few studies that combine writing process and temporal analysis, which mostly focused on think aloud procedures and offline measurements. These have been criticized as being inaccurate representations of the underlying writing process [16].

Therefore, our main aim is to examine students' writing processes and their relationship to writing quality using keystroke logs and temporal analysis. We use an innovative writing technology platform that assists in discriminating between writing phases, mainly planning and writing, by providing separate writing sections to students [17]. The temporal analysis provides insight about how the effect of each of those phases and keystroke log features on students' writing quality may change over time. This can support educators to make judgments regarding students' writing processes and change the ways they teach writing skills.

2. LITERATURE REVIEW

2.1 Writing Research

A common approach of writing research is to consider three phases to describe the writing process: pre-writing, writing, and post-writing [4]. Pre-writing is composed by planning the content of the text to be written. Writing is composing the ideas and transcribing them. Post-writing is revising or reviewing the written text or plan. For simplicity purposes, in our study these phases are referred to as planning, transcribing, and revising, respectively. Efforts in writing research have been made to identify behavioral features that could be an indication of these phases. Early writing research has heavily relied on self-report methods, such as think aloud protocols [3], to examine students' writing process. Over the recent decades, purpose-built software for writing research were developed which collected all information possible during the writing process. Initially these initiatives were restricted to laboratories, but recent advances in software development has now released such software naturalistic settings, allowing for researchers to examine the writing process in real educational environments [17].

2.2 Keystroke Logs in Writing Research

A key research area uses keystroke logging to characterize the writing process. Efforts here have focused on investigating the distributions of various kinds of pauses (e.g., inter-key, intra-word) and their relationship with writing quality. Among the existing studies, [7] found the exponentially modified Gaussian distribution a good fit for inter-key pauses and mapped specific pauses to

Donia Malekian, James Bailey, Gregor Kennedy, Paula de Barba and Sadia Nawaz "Characterising Students' Writing Processes Using Temporal Keystroke Analysis" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 354 - 359

planning process in case they were identified to be long enough. In other work by [1], a mixture of lognormal distribution was used to describe pause pattern in students' inter-key and intra-word pauses. The parameters of the distribution were found to be correlated with writing score. Finally, [9] estimated the distribution of the inter-key and intra-word pause durations for each student and found the best fit to be a heavy-tailed probability distribution called a stable distribution. Because of nearly identical estimates for intra-word and inter-key pauses, they focused on only analyzing intra-word pauses. They found the estimated parameters for each student to be a strong predictor of final score utilizing a linear regression model.

Although these studies provide informative (predictive) models characterizing students' writing process, the effect of isolated phases of writing process have not been evaluated in these models. In addition, even though the use of simple models of regression and correlation analysis makes the interpretation of results easier, the use of more complex models may capture further information about the interrelationships between the extracted characteristics.

Another direction of research focused on using keystroke logs to more comprehensively characterize the phases comprising the writing process [9, 18]. In a study by [2] on modelling students' writing process, some measures were defined to model pauses, bursts and revisions. A burst is defined as the sequence of fast text production and can be identified based on the production of text between two pauses [2]. They suggest that long pauses may reflect planning, as the writers are more likely to have short but well-formed bursts of writing afterward.

Overall, extracted characteristics from keystroke logs in terms of burst, revision summaries and the pattern of pauses, provide important information regarding the underlying writing process. Association of each extracted characteristic with writing quality has been mainly considered as the metric for evaluating the usefulness of the feature [9, 18]. A major challenge is defining meaningful summaries from the writing keystrokes that represent a specific phase of students' writing process as much as possible. To address this problem, we use an innovative writing technology that more explicitly discriminates between the planning and writing phases, by providing separate writing sections to students [17].

Additionally, it is important to know not only which writing phases are relevant to successful writers, but when and in what order they engage with these phases. An approach to temporal data processing has been the aggregation of data in multiple consecutive episodes, so all participants have a similar number of observations [3]. They suggested that the relation between specific aspects of the process with writing quality varies as students' progress in their writing. For instance, the correlation of structuring activities and writing quality is highest at the start of the writing and is lower toward the end [3]. The importance of the temporal analysis of the writing process has been emphasized by several studies [16], however the data representation mostly relies on think aloud procedures and offline measurements that have been criticized as being inaccurate representation of the underlying writing process [16].

Although the importance of taking the moment(s) at which specific aspects of the writing process occur has been emphasized, this is not a dominant view in keystroke logging analysis.

2.3 Current Study

Our first aim is to characterize students' writing processes in terms of a set of features demonstrating the isolated phases. For this purpose, we focus on students' keystrokes characteristics while taking notes, writing the main body of the essay, and organizing

references (each separately). Utilizing a machine learning model, we examine whether section specific characteristics are more predictive of writing quality compared to the characteristics extracted from the overall keystrokes. We also consider adding more features that estimate burst and revision behavior, as well as features representing the extent to which specific aspects of writing process were used during the writing. To evaluate how influential each feature is for each student's writing quality, we adopt a method called SHAP [13] that describes the importance of each feature on the model's prediction for each student. The second aim of the paper is to provide a temporal analysis which helps us to understand whether the importance of features varies over time or remains stable. We address this problem by breaking down students' keystrokes into several writing episodes and comparing the influence of each feature on writing quality across them.

Overall, in this study the following research questions are explored:

1. Do models that characterize the overall writing process miss informative features associating with particular phases of writing? Can we define new characteristics (features) from students' keystrokes to improve these models?
2. How predictive of writing quality are the extracted features at different times? Do we see evidence that the importance of features varies with time?

Our results highlight that characterizing students' keystrokes separately while writing, taking notes and organizing references is more predictive of their writing quality. Additionally, based on our findings, the importance of several features on writing quality changes over time. Investigation of the influential features for individual students clarified the non-linear relationship between some features and writing quality that confirms there exists considerable overlap and interaction between writing phases [4].

3. METHODS

3.1 Participants and Context

The study involves 107 students from the University of Melbourne who enrolled in a business undergraduate course. Students were asked to use a specific online word processing software called Cadmus to write a 1000-word essay as a part of their course, worth 10% of their final mark. Students had to choose between two topics and had 19 days to complete the essay. The performance was marked by teaching staff using a score between 0 to 100.

Cadmus has similar features to other word processing software tools such as body section for writing (body section), editing, highlighting, and additional features such as dedicated sections to take notes (note taking section), and to organize the reference materials (reference section) as well as a single paste restriction of 90 words. Cadmus records every keystroke in each section via the keyboard while students work on their assignment. A more detailed description of this software can be found in [17].

3.2 Data Processing

We next describe the procedure undertaken to characterize students' writing processes by engineering a set of features from keystroke logs. We also processed two concepts of *writing quality* and *writing episode* to assist on answering the research questions.

(A preliminary analysis revealed 4 students had less than 600 words in their essay. They were removed from further analysis.)

3.2.1 Pauses

In this study our focus is on inter-key pauses (the duration between successive keystrokes) that are more likely to be associated with

such processes as deliberation, text planning, reviewing the written text [9]. As [9] suggested, there is a tendency for inter-key pause durations to follow a stable distribution. They fitted this distribution to pause duration data to obtain an informative estimation of the related parameters for each student. We follow a similar procedure as [9]; however, we summarize pauses in each writing section of our dataset separately. This section-specific summarisation could reveal more explicit information regarding which processes were more likely to be engaged during the pause.

We aim to represent students' writing process while they are working on their essay, thus we decided to ignore the pauses more than 2 hours that meant the student had left the session.

In our dataset, the exploration of the distribution of pause durations for each student in each writing section reveals that there is a tendency for pause duration in each section to follow a heavy tail distribution, that means the majority of the pauses are short but there exist a few long pauses. Believing a stable distribution to be a plausible hypothesis, we fitted this distribution to each student's pause data to obtain an estimate of the related parameters in each section. This distribution needs four parameters (α , β , γ , δ) for the complete description.

- The parameter alpha $\alpha \in (0, 2]$, called the tail index. This parameter gives information about the height of the tails.
- The parameter beta $\beta \in [-1, 1]$, called the skewness parameter. The distribution is symmetric if $\beta = 0$. It is skewed to the right if $\beta > 0$, and to the left if $\beta < 0$.
- The parameter delta $\delta \in \mathbb{R}$, is equal to median. Depending on how heavy the tail is, some extreme part of the data may need to be discarded to have a good estimate of this value.
- The parameter gamma $\gamma > 0$, called scale parameter is a measure of dispersion.

Parameters alpha and beta determine the distribution's shape, while parameters gamma and delta define the scale and location of it. For each student, in each section of our dataset we obtain these four estimated parameters. We also estimate these parameters for the overall keystrokes of each student (irrespective of the specific section) as suggested by [9] and consider them as a baseline for the purpose of evaluation.

3.2.2 Bursts

Burst summaries (i.e. mean length of the bursts) can reveal students' fluency in the transcribing phase of the writing process [18]. In keeping with the majority of the literature [14], we identify bursts by breaking up keystrokes at every pause that have longer than 2 seconds of inactivity. We apply this procedure in the body section of our dataset, where students write the main part of their essay. Considering the bursts in which at least one word is typed, we summarize burst length by two features based on the number of words in a burst: The *mean* and the *standard deviation of burst length* for each student. Additionally, we summarize burst duration by two features of *mean* and *standard deviation of burst duration*.

3.2.3 Revision

In this step, our aim is to isolate the revision phase of the writing process from the transcribing phase. Authors of [5] associated single backspaces to spelling correction which reflect self-monitoring, and multiple backspaces to editing in which longer revision occurred. Similarly, we summarize revision at small and large scale with a slight change in the definition; we identify revisions based on the number of word deletions in a writing burst rather than in isolation. We label a burst with single deletion as

small, whereas bursts with multiple deletion as large-scale revision. Two features were extracted to provide measurements related to the revision phase of writing process: The *frequency of both small scale and large-scale revision bursts*.

3.2.4 Time percentage on each writing aspect

To summarize the extent of each specific aspect of writing that was used by each student, we introduce a new set of features, including *the percentage of the total writing time* dedicated to: note taking (total time in note taking section), small- and large-scale revisions (bursts of writing with small or large deletion), transcribing (total time of bursts), and reference organization (total time in reference section).

3.2.5 Writing quality

Writing quality corresponds to the students' final grades on the essay [9, 18]. Previous research has found that students' final grade may not be a reliable measure of success, due to variations in grading essay writing by raters [11]. To account for this variability, we map the students' writing quality to two categories - high and low level instead of the exact grade. In our dataset, the distribution of students' grades lies within the range 60 to 95. We adopt the median value as a threshold for this mapping which is 80. Based on this value we have 40 and 67 students having high quality and low quality writing respectively.

3.2.6 Writing episode definition

One approach to temporal data processing is the analysis of data in multiple episodes to ensure all participants have a similar number of observations [3]. In our study, students were asked to write a 1000-word essay, which provides a good criterion for defining the fixed observations. We define the episodes based on the keystrokes used to complete the fixed number of words in the essay. This way, we have meaningful episodes recording a specific draft of writing.

We split students' writing data into n writing episodes, $\{E_1 + E_2 + \dots + E_n\}$, each of which records students' keystrokes used from the start of writing to the completion of $n*k$ words of the essay. We define 5 writing episodes, each of which involves all the keystrokes from the start of writing to the completion of $n*200$ words. There is no theory for defining the number of episodes and the results may differ based on the choice of this number.

4. Data Analysis

We conducted two sets of analyses: one for each research question.

4.1.1 Research Question 1.

To answer whether models that characterize the overall writing process miss informative features associated with certain phases of writing, we compare the performance of a machine learning model trained on the pause related features extracted from overall keystrokes (*baseline* feature set) to the performance of a model trained on pause features of each writing section separately (*section-specific* feature set). Work in [9] also identified the *total time on task*, along with the pause related features as a strong predictor of writing quality. Thus, we include this feature in the *baseline* and *section-specific* feature sets. Our evaluation is based on the prediction performance of writing quality.

Then, we examine whether we can define new characteristics from students' keystrokes to improve our model. For this purpose, we evaluate our model by adding further features of burst and revision summaries (explained in section 3.2.2 and 3.2.3), as well as the features representing the extent to which specific aspects of the writing process were used (section 3.2.4). We refer to them as *combination* feature set. We utilised XGboost classifier [6] for the

prediction models. Even though this classifier generally provides a good prediction power compared to simple models of regression, understanding what the contribution of each feature were seems to be hard due to the complexity of the model. To evaluate and derive the influence of each feature on writing quality, we use SHAP (SHapley Additive exPlanations) algorithm which can be used to explain the output of any machine learning model [13]. In this algorithm the contribution of a feature is calculated by comparing what a model predicts with and without that feature. Every individual is assigned a SHAP value for each feature that determines the feature's contribution for a change in the model's prediction.

4.1.2 Research Question 2.

To examine how predictive of writing quality are the extracted features at different times, and how their contribution may vary, we broke down students' keystrokes into n episodes from each of which the predictive features (*combination* feature set) were extracted. Then for each episode the predictive power of the features on writing quality was evaluated using the XGBoost model. To identify and compare the contributing features to the models' prediction in each episode, SHAP algorithm was utilised.

5. RESULTS

5.1 Results for Research Question 1

First, we report and compare the prediction power of the introduced feature sets on writing quality. Then we derive and discuss the contribution of each feature on prediction.

5.1.1 Examining the prediction power of feature sets on writing quality

For each set of features a model was trained using leave one out cross validation (with 5-fold nested cross-validation for parameter optimization). The performance of each model on the prediction of writing quality is reported in Table 1 based on the metrics of accuracy and the area under the ROC curve (AUC) [12]. The accuracy and AUC were slightly higher for the *section-specific* feature set compared to the *baseline* set. This could indicate that characterizing the overall process irrespective of the isolated phases may miss specific moments where the features associated to certain writing phase become more important. Adding burst and revision summaries, as well as the time dedication features (*combination* feature set), to the model, improved the performance of the prediction significantly. The *combination* feature set obtained the highest prediction power implying that the introduced features were a better representation of the students' writing quality.

Table 1: Prediction power of each feature set on writing quality based on accuracy and AUC

Features set	Accuracy	AUC
Baseline	70.09	70.63
Section-specific	71.03	71.75
Combination	81.31	82.72

5.1.2 Examining the contribution of features on prediction

The next step was to evaluate the influence of each feature on predictions. For this purpose, we built an XGBoost model on the *combination* feature set (using 5-fold cross validation for parameter optimization). The model was then passed to the SHAP algorithm to explain the influence of each of features on the model's prediction for each student. Since we get individualized explanations of every feature for every student (based on the SHAP values), we can plot the distribution of the importance of each

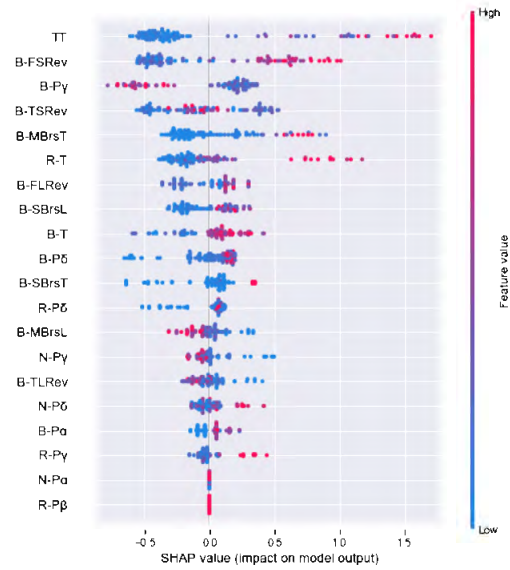


Figure 1: SHAP values for each feature indicating the influence of that feature on writing quality of each student. TT refers to the total time on writing. R, B, N before each feature name refer to Reference, Body and Note sections respectively. α , β , δ , γ refers to pause parameters. TSRev, TLRev refers to the total time on large- and small- scale revisions respectively. FSRev, FLRev refers to the frequency of small- and large- scale revisions. T refers to the total time on specific section. MBrstT, SBrstT are mean and standard deviation of burst time. MBrstL, SBrstL refer to mean and standard deviation of burst length.

feature on the model's prediction, as is presented in Figure 1. The features are sorted by the mean of absolute SHAP values over all students to gain a global insight into the most influential features across all students. We observe the most influential features (globally) were total time on task (TT), frequency of small revisions (B-FSRev), and estimated gamma parameter from pauses in body section (B-P γ) respectively, while some of the pause parameters estimated in the note taking and reference sections (i.e. N-P α , R-P β) had the lowest contribution. For convenience of viewing, only the top-20 features that were globally influential are presented in the figure. In this figure, each row corresponds to a feature and every student has one dot on each row. The x position of the dot is the impact of that feature on the model's prediction for the student (SHAP value), and the color represents the value of that feature for the student (red high, blue low). This reveal, for example, that a high percentage of time on the reference section (R-T) increases the chance of having high quality writing for a subset of students (red dots in the R-T row, and on the right side of the plot).

Below is our interpretation of some of the features detected as important by the prediction model. It is worth mentioning that this interpretation is intended for high-level model interpretation, and the model's decision making was more complex and took the interaction between features into account.

Total time on task (TT) was found to be the strongest predictor of writing quality. A subset of students (red dots on the related row of figure) who spent a lot more time than others on the essay are more likely to produce a high-quality writing. This supports previous research [9] and could be an indication of student's motivation in completing the writing task [10]. However, there are also students with lower time spent on task, but a higher chance of producing high-quality writing (blue dots on the right side of the plot in the

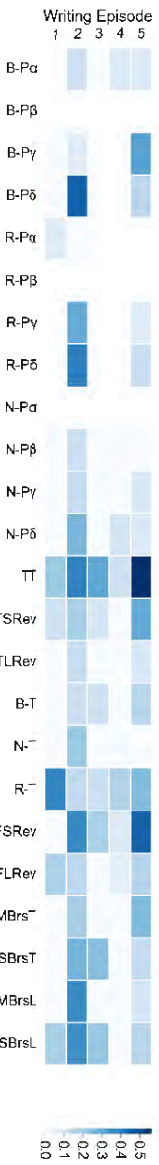


Figure 2: Visualization of the global importance of each feature on the prediction in each writing episode based on the SHAP values. Darker colors indicate higher contribution of that feature on prediction in that episode.

related row). This could indicate other features are impacting the contribution of this feature.

The coloring of the second most important predictor - the percentage of time on small revision ($B-FSRev$) shows us how a higher frequency of small revision means a higher the chance of producing high quality writing for most students. It could imply that the writing quality improves if revisions are performed more frequently. This is in agreement with previous research [15] suggesting that good writers stop their writing more often to perform revision and to correct their spelling and grammatical errors as they write compared to weak writers.

The estimated gamma and alpha parameters from pauses inside the body section ($B-P\gamma$, $B-P\delta$) shows a (negative, positive) association with the chance of producing a high-quality writing for most of the students. Together these parameters mean lower variation of pause durations without having very large pauses. This could be an indication of steadiness and fluency in writing and its direct relation with writing quality which is in agreement with previous research [9]. Percentage of time on reference section ($R-T$) also shows a positive effect on writing quality for several students. Again, we observe that this argument does not hold true for several students.

The higher percentage of time on transcribing ($B-T$) increase the chance of having high-quality writing. One alternative justification could be found in a study by [8]. They found that writers who spend most time on other aspects of writing such as planning tend to dislike writing and this may lessen the text quality.

Estimated delta and gamma parameters from pauses in the note taking section ($N-P\delta$, $N-P\gamma$) shows a (positive, negative) association with the chance of producing a high-quality writing for most students. Together these parameters mean more large pauses and lower variation in pause duration. This means that steadily taking large pauses while taking notes leads to a higher chance of high-quality writing. Based on a work by [2], large pauses may reflect planning process. Since this pattern is observed in the note taking section of our dataset, we could more certainly connect these steady long pauses to thinking periods on planning.

Even though there are several features such as percentage of time on small revision ($B-TSRev$), mean burst length (B_MBrsL), estimated gamma from reference and body section ($R-P\delta$, $B-P\delta$), that are detected as influential, we cannot observe their clear linear association with writing quality. For a subset of students, the higher value is associated with higher quality and for others it is the opposite. Again, this indicates that the importance of these features is impacted by other features.

5.2 Results for Research Question 2

In this section we answer the second research question in two steps.

5.2.1 Examining the predictive power of features on writing quality over time

Using the XGBoost classification algorithm, we examined the predictive power of the extracted feature in our study (*combination*

feature set) on writing quality at each writing episode. We report our result using the metrics of accuracy and area under the ROC curve (AUC), based on leave one out cross-validation (with 5-fold hyper parameter optimisation). The outcome is shown in Table 2 demonstrating a high and, in some episodes, moderate success rate in predicting students' writing quality (better than random chance).

Table 2: Prediction power of extracted features in each writing episode based on the accuracy and AUC

Episode#	Accuracy	AUC
1	69.16	70.26
2	67.29	67.12
3	69.16	66.01
4	79.43	79.22
5	81.31	82.72

5.2.2 Examining the contribution of features on prediction over writing episodes

Our next step was to examine the contribution of each feature on the prediction of writing quality at each writing episode and examine whether the contribution varied. For this purpose, an XGBoost model was trained based on the *combination* feature set in each episode (using 5-fold cross validation for parameter optimization). Each model was then passed to the SHAP algorithm to explain the importance of each of feature on the prediction. The result is reported in Figure 2, in which the global contribution of each feature in each writing episode is visualized based on the mean of absolute SHAP values for that feature over all students.

A darker color indicates a higher contribution of that feature on the prediction model in that episode. We see the contribution of features on writing quality are quite different across the writing episodes. Although the importance of total time on task (TT) is relatively stable over all the episodes, the importance of the note taking related features such as percentage of time on note taking section ($N-T$), and estimated pause parameters ($N-P\delta$, $N-P\gamma$) were found to be more influential in the 2nd writing episode. The patterns of pauses in the reference section (i.e. percentage of time on the reference section ($R-T$), as well as the estimated gamma and delta parameters ($R-P\gamma$, $R-P\delta$) also show stronger influence on the prediction of writing quality at the beginning of writing.

Our temporal analysis reveals that the importance of writing behaviour on writing quality may change over time. Thus, characterizing students' writing process irrespective of time may hide the effect of meaningful and predictive writing behaviors. Moreover, this analysis could be used to predict students writing quality over time and act as a filter for early targeting of students with different writing qualities opening up feedback opportunities.

6. DISCUSSION

This study was based on a fully online web authoring tool called Cadmus. The availability of separate sections for body text, note taking and referencing allowed us to separately extract the

keystroke logs of different sections. From these logs we were able to engineer multiple sets of features capturing different aspects of students' writing processes ranging from patterns of pauses, burst and revision summaries as well as the time dedicated to specific aspects of the writing process. We compared the performance of a model trained on the pause pattern related features extracted from overall keystrokes (*baseline*), to the performance of a model trained on pause pattern related features of each writing section separately. The section-specific model performed slightly better. This indicates that the baseline model may miss or "overlook" on specific moments where the features associated with certain writing phases become more important. The performance of the section-specific model was further improved by adding more features including burst and revision summaries and percentage of time on specific activities.

The feature importance of the resultant model is visualized for every student showing how specific behaviour that have positive effect on writing quality for one student may have negative effect for another because of the impact of other features. This is consistent with a theory of writing which suggests there exists considerable interaction and overlap between writing phases [4]. This also emphasises the necessity of using models that capture the interrelationship between features rather than simple correlation and regression analysis.

The current study also examined whether the influence of extracted features on writing quality varies during specific moments of writing. Based on our results, the influence of several features varied across writing episodes indicating the importance of taking the temporal aspects of writing process into account. Through this study, we hope to develop a better understanding of students' writing process in authentic educational settings.

More detailed results, discussions, and additional figures that could not be included in this version of the paper for the reason of space, is available in the longer version of the paper on this [link](#).

6.1 Limitations and Future Work

The first limitation of this study is the generalizability of the interpretations regarding the influential features. This study was based on the essay writing of undergraduate students with diverse writing and language backgrounds. The influential features might differ for data from a more diverse set of students and across variations in topic, genre and prompts. The next limitation arises from considering all the activities in the note taking and reference sections to be associated with the related phase of writing. This association is irrespective of the actual written text. For the next study we aim to consider the actual text entered in each section. This may help to distinguish between weak and strong planning behaviour and the related impact on writing quality.

REFERENCES

- [1] Almond, R., Deane, P., Quinlan, T., Wagner, M. and Sydorenko, T. 2012. A preliminary analysis of keystroke log data from a timed writing task. (*ResearchReportNo.RR-12-23*). (Dec. 2012), Princeton, NJ: Educational Testing Service. DOI:<https://doi.org/10.1002/j.2333-8504.2012.tb02305.x>.
- [2] Baaijen, V.M., Galbraith, D. and de Glopper, K. 2012. Keystroke analysis: reflections on procedures and measures. *Written Communication*. 29, 3 (Jul. 2012), 246–277. DOI:<https://doi.org/10.1177/0741088312451108>.
- [3] den Bergh, H. and Rijlaarsdam, G. 1996. The dynamics of composing: modeling writing process data. *The Science of Writing: Theories, Methods, Individual Differences, and Applications*. New York: Lawrence Erlbaum Ass. 207–232.
- [4] Biggs, J. 1988. The role of metacognition in enhancing learning. *Australian Journal of Education*. 32, 2 (Aug. 1988), 127–138. DOI:<https://doi.org/10.1177/000494418803200201>.
- [5] Chanquoy, L. 2009. Revision processes. *The SAGE Handbook of Writing Development*. London: SAGE Publications Ltd. 80–97. DOI:10.4135/9780857021069.n6.
- [6] Chen, T. and Guestrin, C. 2016. XGBoost: a scalable tree boosting system. *Proc. of the 22nd Conference on Knowledge Discovery and Data Mining* (New York, USA, Aug. 2016), 785–794. DOI:10.1145/2939672.2939785.
- [7] Chukharev-Hudilainen, E. 2014. Pauses in spontaneous written communication: a keystroke logging study. *Journal of Writing Research*. 6, 1 (Jun. 2014), 61–84. DOI:<https://doi.org/10.17239/jowr-2014.06.01.3>.
- [8] Green, D.W. and Wason, P.C. 1982. Notes on the Psychology of Writing. *Human Relations*. 35, 1 (Jan. 1982), 47–56. DOI:<https://doi.org/10.1177/001872678203500104>.
- [9] Guo, H., Deane, P.D., van Rijn, P.W., Zhang, M. and Bennett, R.E. 2018. Modeling basic writing processes from keystroke logs. *Journal of Educational Measurement*. 55, 2 (Jun. 2018), 194–216. DOI:<https://doi.org/10.1111/jedm.12172>.
- [10] Hayes, J. and Gradwohl Nash, J. 1996. On the nature of planning in writing. *The Science of Writing: Theories, Methods, Individual Differences, and Applications*. Hillsdale, NJ, US: Lawrence Erlbaum Associates, Inc. 29–55.
- [11] Kayapinar, U. 2014. Measuring essay assessment: intra-rater and inter-rater reliability. *Eurasian Journal of Educational Research*. 14, 57 (Oct. 2014), 113–135. DOI:<https://doi.org/10.14689/ejer.2014.57.2>.
- [12] Ling, C.X., Huang, J. and Zhang, H. 2003. AUC: a statistically consistent and more discriminating measure than accuracy. *Proc. of the 18th Conference on Artificial Intelligence* (San Francisco, CA, USA, Aug. 2003), 519–524.
- [13] Lundberg, S.M. and Lee, S.-I. 2017. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems 30* (Long Beach, Ca, US, Dec. 2017), 4765–4774.
- [14] Rosenqvist, S. 2015. *Developing pause thresholds for keystroke logging analysis*. B.A. thesis. University of Umea, Sweden. Retrieved from <http://umu.diva-portal.org/smash/get/diva2:834468/FULLTEXT01.pdf>.
- [15] Stallard K.C. 1974. An analysis of the writing behavior of good student writers. *Research in the Teaching of English*. 8, 2 (Summer. 1974), 206–218.
- [16] Tillema, M., van den Bergh, H., Rijlaarsdam, G. and Sanders, T. 2011. Relating self reports of writing behaviour and online task execution using a temporal model. *Metacognition and Learning*. 6, 3 (Dec. 2011), 229–253. DOI:<https://doi.org/10.1007/s11409-011-9072-x>.
- [17] Trezise, K., de Barba, P.G., Jennens, D., Zarebski, A., Russo, R. and Kennedy, G. 2017. A learning analytics view of students' use of self-regulation strategies for essay writing. *Proc. of the ASCILITE 2017* (Toowoomba, QLD, AUS, Nov. 2017), 411.
- [18] Zhang, M., Hao, J., Deane, P. and Li, C. 2018. Defining personalized writing burst measures of translation using keystroke logs. *Proc. of the 11th Conference on Educational Data Mining* (Buffalo NY, Jul. 2018), 549–552.

Skills Embeddings: a Neural Approach to Multicomponent Representations of Students and Tasks

Russell Moore*, Andrew Caines*, Mark Elliott*, Ahmed Zaidi*, Andrew Rice* and Paula Buttery*
ALTA Institute

*Computer Laboratory †Cambridge Assessment
University of Cambridge

{rjm49 | apc38 | mwe24 | ahz22 | acr31 | pjb48}@cam.ac.uk

ABSTRACT

Educational systems use models of student skill to inform decision-making processes. Defining such models manually is challenging due to the large number of relevant factors. We propose learning multidimensional representations (embeddings) from student activity data – these are fixed-length real vectors with three desirable characteristics: co-location of similar students and items in a vector space; magnitude increases with skill, and that absence of a skill can be represented. Based on the Multicomponent Latent Trait Model, we use a neural network with complementary trainable weights to learn these embeddings by backpropagation. We evaluate using synthetic student activity data that provides a ground-truth of student skills in order to understand the impact of number of students, question items and knowledge components in the domain. We find that our data-mined parameter values can recreate the synthetic datasets up to the accuracy of the model that generated them, for domains with up to 10 simultaneously active knowledge components, which can be effectively mined using relatively small quantities of data (1000 students, 100 items). We describe a procedure to estimate the number of components in a domain, and propose a component-masking logic mechanism that improves performance on high-dimensional datasets.

Keywords

knowledge representation, skills embeddings, multicomponent latent trait model

1. INTRODUCTION

Intelligent tutoring systems (ITS) are required to make decisions about which tasks to present to which students. Thus they should be equipped with objective, accurate models of student skillsets, to inform these choices. Student activity logs are a source of information for such models, which could be built by hand-crafted feature extraction, or using data mining methods. We explore the latter, using machine-learning of fixed-width multidimensional representations.

Russell Moore, Andrew Caines, Mark Elliott, Ahmed Zaidi, Andrew Rice and Paula Buttery "Skills Embeddings: a Neural Approach to Multicomponent Representations of Students and Tasks" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 360 - 365

tations. We call these SKILLS EMBEDDINGS, after *word embeddings* – vector representations of words and language constructs that have allowed dramatic advances in the natural language processing field [6, 8].

With word embeddings, semantically similar words are positioned near each other in a latent vector space: *e.g.* ‘boat’ should be nearer to ‘car’ than ‘politics’, based on natural language data. We transfer the vector space idea to skills. For the ITS scenario, we seek specific desirable traits:

- Skills embeddings are **co-proximal** in the vector space if they represent entities comprising similar skills.
- Embedding **magnitude** grows with skill – specifically, a higher skill level should entail a larger value within the embedding. This lets us track skill gain intuitively.
- It should be possible to represent the special case where a skill is absent. We will refer to this characteristic as **skill masking**.

We would also ideally like to be able to **detect dimensionality**: the number and dependency structure of skills within the domain should not need to be specified in advance.

In this work, we propose an artificial neural network – based on the Multicomponent Latent Trait Model [12] – to learn skills embeddings. We train it using synthetic student activity datasets, with a varying number of skills in the domain. We show that the embeddings exhibit our three desired characteristics, and present a procedure to cater to the fourth.

2. BACKGROUND

Our work relies on some core principles about the nature of knowledge domains, the way that student ability and item difficulty interact, and the idea that knowledge acquisition can be traced in student activity logs [5].

2.1 Knowledge components

For any given educational domain, such as physics, mathematics or language learning, we can break domain-specific knowledge down into atomistic units known as KNOWLEDGE COMPONENTS (KCs), as described by Koedinger *et al.* [4]. We treat these as synonymous with ‘skill’ where the skill is irreducible – if skill C comprises irreducible subskills A and B, we do not represent C, but treat co-occurrence of A and B as the pattern for C. We think of a subject domain as a

set K of KCs to be acquired. In our datasets, domain size $|K| \in [1, 100]$.

2.2 Rasch model

The Rasch item response model [11] is a well-known formulation for the success probability of student s attempting item (question) i , derived by transforming the difference between student ability θ_s and item difficulty β_i through a sigmoid function. That is, the probability of success is given by:

$$Pr(X_{si} = 1 \mid \theta_s, \beta_i) = \sigma(\theta_s, \beta_i) \quad (1)$$

where σ is the standard logistic sigmoid function:

$$\sigma(\theta_s, \beta_i) = \frac{1}{1 + \exp(-(\theta_s - \beta_i))} \quad (2)$$

Note that an evenly-matched student-item pair has $\theta = \beta$ and a pass-rate of 0.5. The Rasch model assumes a single dimension of proficiency and embodies *invariant comparison* – this means the student parameter θ can be eliminated algebraically during estimation of the item parameters β , and vice versa [9, 14]. This principle allows Rasch items (and by extension our embeddings) to stand alone as objective representations, independent of the conditions in which they were measured.

2.3 Multicomponent Latent Trait Model

The Rasch model can be extended to the MULTICOMPONENT LATENT TRAIT MODEL (MLTM) of Whitely [12]. Here, the scalars θ and β are replaced by vectors, and the result is a product of sigmoids. The formulation is as follows:

$$Pr(X_{si} = 1 \mid \theta_s, \beta_i) = \prod_{k \in \text{skills}(i)} \sigma(\theta_{sk}, \beta_{ik}) \quad (3)$$

Hence the act of student s successfully passing item i is modelled as the conjunction of successes at each of the item’s problem-solving steps (denoted k). The probability is given by the product of the probabilities of completing the steps and each step behaves as a Rasch model whose parameters are the corresponding elements of θ_s and β_i .

2.4 Item calibration

Traditionally, item calibration with Rasch-type models is carried out using the Birnbaum iteration [2]. However, the Birnbaum algorithm is one-dimensional and to the best of our knowledge has not been extended to multiple dimensions. Moreover, the Rasch approach does not readily allow for the absence of skills: parameters would have to be set to $-\infty$, which is impractical for data mining, particularly in cases where the skill is absent both from a question and student’s representations¹

2.5 Q-Matrix

The Q-MATRIX [10, 1] is a binary- or probability-valued matrix that describes which skills are required for particular tasks. This has been used previously, for instance, in the Linear Logic Test Model [13]. Each column of \mathbf{Q} represents a task/item i , and each row a component $k \in K$.

¹Even assuming infinite arithmetic is allowable, if θ and β are both $-\infty$ then $(\theta - \beta) = 0$ and the probability of success is calculated as 0.5; in fact, for an unrequired skill, it should always be 1, since an unrequired step is always ‘passed’.

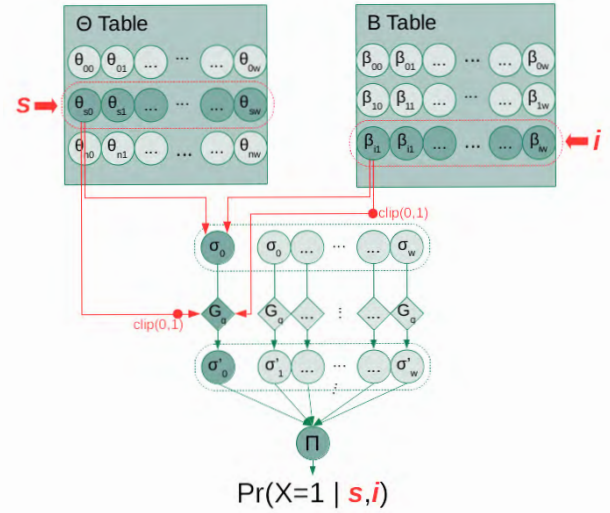


Figure 1: Neural network architecture

The (binary) Q-matrix for a curriculum of items is as follows:

$$q_{ik} = \begin{cases} 1 & \text{if } k \in \text{skills}(i), \\ 0 & \text{else;} \end{cases} \quad (4)$$

The Q-matrix for students is similar, with element q_{sk} representing the ability of student s at skill k .

3. DATA

Our datasets follow a *summative* assessment scenario: the item bank is a static test, to be attempted by many students. Each student attempts all items only once. Each response is dichotomous: either right ($X=1$) or wrong ($X=0$).

Student activity data is synthesised using a statistical model whose probability mass function (pmf) is just equation (3). The response x_{si} (of student s on item i) is determined by ground-truth values of the MLTM parameters, θ_s^* and β_i^* . These are the targets we hope to recover from the dichotomous outcome data: we want our embeddings to converge on these values.

The elements of θ_s^* and β_i^* themselves are generated uniformly randomly for each student and item – their minimum and maximum values are chosen by an earlier randomised search process, that looks for suitable bounds to generate a balanced dataset given a specific dimensionality $|K|$.

We generated datasets with dimensionalities $|K| \in \{1, 2, 5, 10, 100\}$. We also created datasets where only a subset of components are active – for these datasets, the active components are chosen at random. We use $|I|=100$ items and $|S|=1000$ students.

4. IMPLEMENTATION

In this section we describe the neural network, including its design features, software used, and training approach.

4.1 Neural network architecture

The neural network in this work is a binary classifier. In a normal supervised learning task, a classifier takes an input set of features $\Phi(x)$ and class label C and learns the probability that a datapoint x is in a given class: $Pr(x \in C \mid \Phi(x))$.

In the embedding generation task there is no feature mapping Φ . Instead we have datapoints of form $(s, i, pass \in \{T, F\})$, describing whether student s passed item i . Inputs to the neural network are s and i , and class label $pass$.

The ‘features’ themselves are learned internally, with two distinct sets of trainable weights. One set of weights ($\Theta \in \mathbb{R}^{|S| \times |K|}$) represents the students, the other ($B \in \mathbb{R}^{|I| \times |K|}$) the items. Whenever s occurs in a datapoint, the weights for s (synonymous with θ_s) are selected from the table, and the same happens for β_i when i occurs.

The weights are fed into a locally connected layer that represents the components of the MLTM. Each unit in the layer applies a sigmoid function to generate a per-component probability. The component probabilities are then multiplied to get the overall output probability. This is scored against the true $pass$ value using a loss function, and the error is backpropagated to the weights tables. The weights are re-used whenever s or i appear in a datapoint, so they are forced towards values which best fit all observations. The trained rows of weights serve as our fixed-width embeddings.

The architecture is illustrated in Figure 1. For clarity, the connections are shown only for a single component, but all components function in parallel in the same way. The diamonds on the diagram represent Q-gates, trainable logic components which we will now describe.

4.2 Q-gates – logic for absent components

In high-dimensional domains, items usually do not exercise all skill components simultaneously. For instance, in both assessment and instruction, questions are usually designed to focus on a subset of skills. Rather than model the subset of skills explicitly, we iterate across the whole domain K and let a logic layer selectively deactivate components:

$$Pr(X_{si} = 1 \mid \theta_s, \beta_i) = \prod_{k \in K} G_q(q_{sk}, q_{ik}, \sigma(\theta_{sk}, \beta_{ik})) \quad (5)$$

G_q is a *Q-gate*, a ternary logic gate related to logical implication, with the following truth table for each component per student, q_{sk} , and per item, q_{ik} :

q_{ik}	q_{sk}	G_q
1	1	$\sigma(\theta_{sk}, \beta_{ik})$
1	0	0
0	1	1
0	0	1

Q-gates are implemented as part of the neural network, and modify the component-level sigmoid outputs:

$$G_q(q_{sk}, q_{ik}, \sigma(\theta_{sk}, \beta_{ik})) = \sigma(\theta_{sk}, \beta_{ik}) q_{ik} q_{sk} + (1 - q_{ik}) \quad (6)$$

The correct values for q_{ik} and q_{sk} are learned during training. These *q-values* can either be stored in their own set of weights, or represented by special values in θ and β . We use the latter technique with *weight clipping*, explained next.

4.3 Weight clipping

We employ *weight clipping* for two reasons: to ensure component positivity (so that vector magnitude must grow with skill), and to implement Q-gates. To ensure components take only positive values, weights are clipped to $[1, W]$, with large W to allow for changes in value during training. The range $[0, 1)$ is reserved to switch the component’s Q-gate.

4.4 Training

The network was trained with a categorical cross-entropy loss function and the Adam optimiser [3]. Generally, training is fast, and a learning rate $\alpha \in [0.01, 0.1]$ is stable. Weight initialisation is significant for convergence speed and fit: a uniform random initialisation over $[\theta_{min}, \theta_{max}]$ for students and $[\beta_{min}, \beta_{max}]$ for items was found to work well. From all instances in the training set, 10% were randomly chosen for validation and to trigger early-stopping on $loss_{val}$ with *patience* = 10 (i.e. we wait for a better value for ten more epochs before quitting, keeping our best weights). This work was implemented in Python 3.6 using Keras with a TensorFlow back-end, and scikit-learn.

5. EVALUATION

In this section we describe how the embeddings were evaluated *vis-à-vis* our desired characteristics.

5.1 Prediction agreement

We attempt to recreate the original datasets by using our embeddings $\hat{\theta}$ and $\hat{\beta}$ to seed our statistical MLTM model. We then score correlation and agreement between the outputs of the original process (seeded with targets θ^* and β^*) and the embedding-seeded process.

Since our dataset is synthetic, we can directly access the probabilities that determine the outcomes, and thus can measure the Pearson’s correlation between these and the predicted probabilities from the embeddings. This is:

$$\rho_{X,Y} = \frac{cov(P^*, \hat{P})}{sd(P^*), sd(\hat{P})} \quad (7)$$

where P^* and \hat{P} are the true and predicted probabilities of a pass, $cov(\cdot, \cdot)$ is covariance and $sd(\cdot)$ is standard deviation.

Because the generator process is stochastic, there will always be some element of chance in the observed outcomes. *Cohen’s Kappa* gives a measure of the agreement beyond chance. For N datapoints, where n_{agreed} is the observed agreement between both runs, and $n_{(k;seed)}$ is the number classed as category k by the model seeded with *seed*:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \text{ where } \begin{cases} p_o = n_{agreed}/N \\ p_e = \frac{1}{N^2} \sum_{k \in \{T, F\}} n_{(k; \theta^* \beta^*)} n_{(k; \hat{\theta} \hat{\beta})} \end{cases} \quad (8)$$

5.2 Co-proximity & Magnitude

To assess our co-proximity requirement, we measure Euclidean distance from the aligned embedding $\hat{\theta}$ (or $\hat{\beta}$) to its target, θ^* (or β^*). We compare this to the mean distance from other vectors in the space, and test for significance to show that co-proximity to target is not merely by chance.

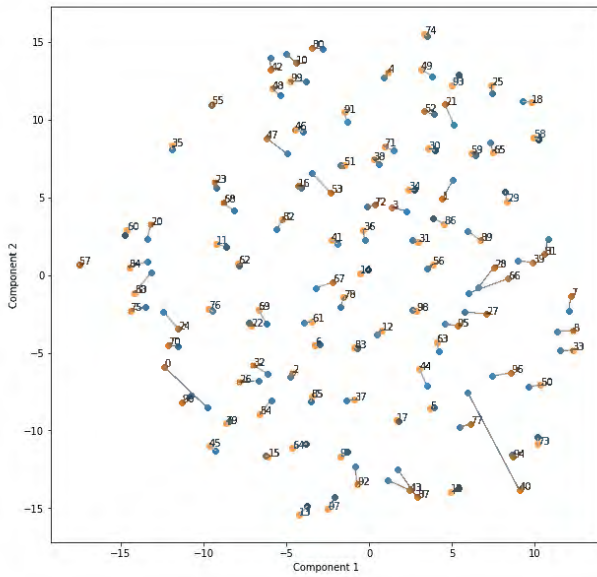


Figure 2: t-SNE visualisation of embeddings in a 10-dimensional space, all active, with 1000 students and 100 items (orange=embedding, blue=target).

To assess magnitude growth with skill, we use Pearson correlation (equation 7) at the component level between the elements of $\hat{\theta}$ (or $\hat{\beta}$) and of target θ^* (or β^*). A strong correlation shows that these values grow together as desired.

5.3 Aligning the components

Embeddings are identifiable only up to the ordering of the components due to conjunctive commutivity: columns in the Θ -Table will be aligned with the B-Table, since they were trained together, but they may be permuted differently to the columns in the original target vectors. To visualise the data, or calculate deviations from the true parameters, we must first align the predicted components. A hill-climbing algorithm can find the order needed to minimise the per-column squared error between the predicted and true values. While not guaranteed to find the global optimum, it is nonetheless reliable. Once the components are aligned, the embeddings can be plotted (after dimensionality reduction such as PCA or t-SNE for $|K| > 2$). The mean absolute parameter errors are given as θ_{MAE} and β_{MAE} in Table 1(b).

Figure 3 shows the mined values for a 2 KC domain, with 1-2 active components: Q-gates allow components to be switched on/off. Dotted lines show the thresholds below which the Q-gate treats a component as inactive. Figure 4 shows the mined values for a 10 KC domain, with 1-3 active components. Here the clustering of target points (blue) are more pronounced than in the all-active data (Figure 2). The embeddings (orange) cluster close to their targets due to the Q-gate mechanism.

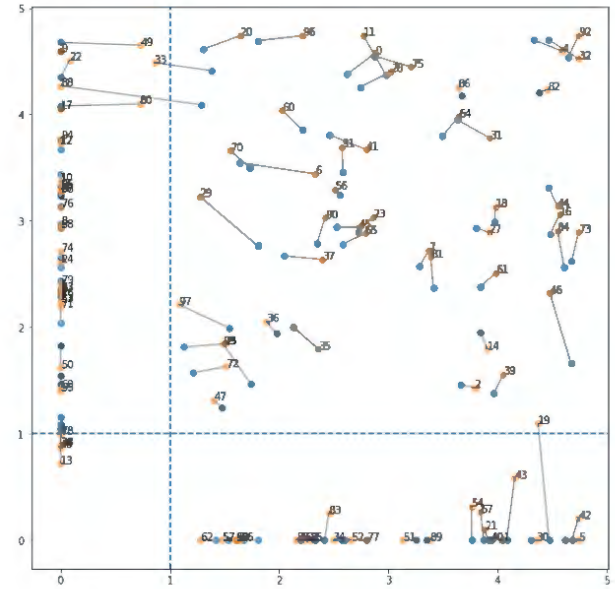


Figure 3: Direct plot of Q-gated embeddings in a 2-dimensional space, 1000 students and 100 mixed items (50×2 -components, 50×1 -component). Dotted lines show the Q-gate activation regions.

6. RESULTS & DISCUSSION

Table 1 gives summary statistics for all-active (upper section) and Q-gated (lower section) datasets.

Table 1(a) gives raw accuracy and Cohen's κ measure agreement between true and predicted outcomes, and Pearson's ρ measures correlation between the underlying probabilities. The reproduction (*repro*) scores for *Acc* and κ show how well the original model can reproduce its own data – this indicates the level of stochastic noise in the target dataset.

Reproduction accuracy drops from 0.844 to 0.651 as domain size increases, but this is more pronounced (0.684 to 0.310) for κ , implying that much of the accuracy score is down to chance, and κ is a more useful measure. At low to moderate dimensions (1a to 10a), mined κ values approach the model's own agreement, but for the high dimension (100a) they do not even achieve half of this limit, although still better than chance ($\kappa = 0$). For Q-gated data (2q2, 10q3, 10q5, 100q5) the embeddings produce higher accuracy and κ throughout than their non-Q-gated counterparts.

Pearson's correlation on the underlying probabilities (unaffected by stochastic noise) is very good (>0.9) for all lower $|K|$ data (with or without Q-gates), but for (100a) this drops to 0.442 – the Q-gated version (100q5) scores 0.754, an improvement of over 70%.

6.1 Co-proximity & Magnitude

Table 1(c) gives mean Euclidean distances of embedding to target, alongside the mean distance to other points in the dataset. Mean and standard deviations are given along with Welch's t-test results. In all cases the mined vectors have significantly ($p < 0.01$) smaller mean distances to target than to other vectors in the space, indicating co-proximity between

Table 1: (a) Accuracy, Cohen’s κ agreement, Pearson’s correlation (ρ). The *repro* scores show how well the original dataset generator agrees with itself between runs, giving an upper limit to the score. (b) Parameter level error scores for students and items. (c) Mean Euclidean distance to target (D_{target}) and to other vectors (D_{others}), with t-test scores.

Model		(a) Model fit			(b) Param. fit.		(c) Co-proximity in vector space		
Name	Dims	Acc (repro)	κ (repro)	ρ^*	θ_{MAE}	β_{MAE}	D_{target}	D_{others}	t^{**}
1a	1	0.844 (0.844)	0.680 (0.684)	0.994	0.29	0.11	0.11 (0.09)	3.66 (0.65)	-36.9
2a	2	0.776 (0.777)	0.550 (0.556)	0.986	0.55	0.46	0.79 (0.38)	3.74 (0.80)	-33.1
5a	5	0.732 (0.742)	0.429 (0.439)	0.961	0.94	0.79	2.51 (1.36)	8.63 (1.24)	-33.0
10a	10	0.721 (0.733)	0.425 (0.463)	0.930	4.61	3.42	13.56 (3.32)	16.47 (1.06)	-8.26
100a	100	0.572 (0.651)	0.151 (0.310)	0.442	7.25	6.36	79.31 (5.40)	88.30 (2.18)	-15.0
2q2	2 (1-2)	0.801 (0.806)	0.576 (0.587)	0.987	0.58	0.41	0.24 (0.17)	4.76 (0.71)	-61.2
10q3	10 (1-3)	0.771 (0.802)	0.561 (0.588)	0.943	1.30	0.13	0.85 (0.63)	9.18 (1.49)	-51.2
10q5	10 (1-5)	0.822 (0.832)	0.503 (0.522)	0.942	1.33	0.19	1.12 (0.84)	10.53 (1.48)	-54.9
100q5	100 (1-5)	0.732 (0.821)	0.509 (0.555)	0.754	3.09	0.27	1.13 (0.84)	10.53 (1.48)	-54.0

*($p < 0.01$) ** (Welch’s t-test, $df=98$, $p < 0.01$)

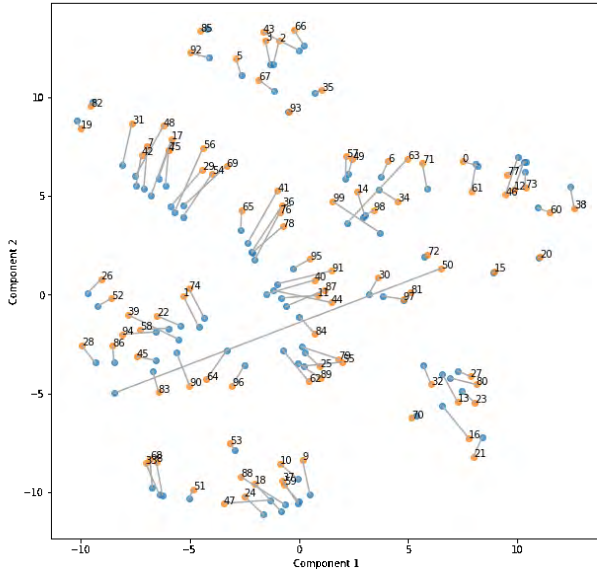


Figure 4: t-SNE visualisation showing 100 items, calibrated from 1000 students. Here, only 1-3 dimensions (from 10) are active for any item. The other dimensions are masked using Q-gates.

mined and true vectors as desired. Again Q-gates have a notable effect: for instance, the quotient (D_{others}/D_{target}) increases from 1.11 for 100a to 9.30 for 100q5.

Component-level correlations for true and mined parameter values (not tabled) are strongly correlated ($\rho \geq 0.90, p = 0.01$) for low to moderate $|K| \in [1, 10]$ showing our second desired characteristic of magnitude growth with skill. As with other results, a weaker positive correlation ($\rho = 0.21, p = 0.01$) was measured for $|K| = 100$.

6.2 Skill masking

Our embeddings achieve a high correlation with target outcome probabilities for all but $|K| = 100$. Similar patterns can be seen with other scores. Overall, larger error in the

vector space (100a in Table 1(b)) seems to contribute to markedly reduced model fit (100a in Table 1(a)).

There are at least two factors at play here: firstly, for large $|K|$, the ‘curse of dimensionality’ makes distance metrics less meaningful, and it becomes difficult to determine distance (or similarity) between vectors. The second factor is informational: if an item has 100 active components, a student must achieve a pass-rate of 99.3% *on each component* to get 50% pass-rate on the item. Very easy components carry little information: our expectation that a student would pass them is almost always met. This manifests as a shallow gradient on the sigmoid in this region ($x = 4.95$), making parameter values very sensitive to stochastic noise in training data. Furthermore, dichotomous results tell us nothing about which component caused a failed attempt. These factors make for a difficult machine-learning task.

Fortunately, the Q-gated datasets show a different pattern. They behave more like low-dimensionality data: for instance, 100 (1-5 active) dimensions – *versus* 100 (all active) – shows both better probability correlation ($0.75 > 0.44$), and component level error ($3.09 < 7.52$ for θ and $0.27 < 6.36$ for β). A similar effect is seen in the 10-dimensional data.

The ability to mask off certain components is useful. For instance, $|K| = 100$ may be a reasonable domain size, but items will often have far fewer active skills, *e.g.* Pardos *et al.* [7] used a question-set for high-school mathematics with $|K| = 105$, but a maximum of three skills per question. Masking is vital to represent such a domain as fixed-width vectors. Moreover, with Q-gates, the exact number and composition of skills per question need not be known: it can be learned during training. Hence the width of our embeddings need not exactly match the domain: if they are too wide, the Q-gates will trim excess components. We give a procedure to estimate $|K|$ next.

6.3 Dimensionality estimation

Although it is not possible to directly detect the dimensionality $|K|$ of a domain, there is a simple procedure to estimate it. Embeddings are trained across a span of candidate values $|K|_{cand}$ and the mean maximum accuracy for each $|K|_{cand}$

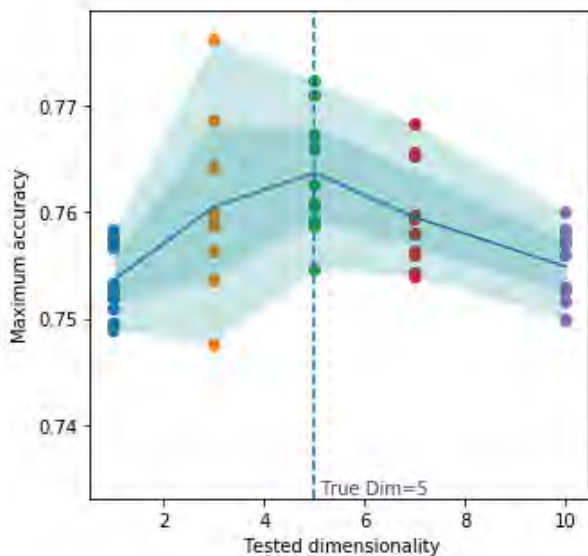


Figure 5: Dimensionality detection – candidate dimensions on x-axis, maximum accuracy for each fit on y-axis; ten runs per candidate, line plot shows mean maximum accuracy.

is calculated. Figure 5 shows the result of this process for a dataset with $|K| = 5$, $|S| = 1000$, $|I| = 100$. We plot candidates $|K|_{\text{cand}} \in \{1, 3, 5, 7, 10\}$ against the mean maximum accuracy (over 10 repetitions) of fit to the dataset. The peak at $|K|_{\text{cand}} = 5$ reveals the true value of $|K|$.

7. FUTURE WORK

We are in the process of mining skills embeddings from a major online physics-teaching platform. We believe the embeddings discovered will reveal more about realistic dimensionality and skill composition, and help us to study changes in student ability over time. We also intend to report on the interpretability of embeddings by human experts.

8. CONCLUSION

This work introduces a new technique to mine SKILLS EMBEDDINGS – student and item vector representations based on the Multicomponent Latent Trait Model – using a neural network with complementary weights. This was applied to synthesised student activity datasets, to recover the original seed parameters. We were able to extract these parameters in moderately high-dimension data ($|K|=10$) even for small datasets (100 items, 1000 students).

We gave four desired characteristics for our embeddings: co-proximity of similar objects in vector space, growth of magnitude with skill, ability to model missing skills, and applicability in domains of unknown dimension. We showed our embeddings support all but the fourth, and gave a procedure to mitigate this. We introduced Q-GATES, a skill masking mechanism that boosts model fit for high dimensional domains with realistic constraints.

9. ACKNOWLEDGMENTS

This paper reports on research supported by Cambridge Assessment, University of Cambridge. We thank the Isaac Physics team, our colleagues in the ALTA Institute, and the three anonymous reviewers for their valuable feedback.

10. REFERENCES

- [1] M. Birenbaum, A. Kelly, and K. Tatsuoka. Diagnosing knowledge states in algebra using the rule-space model. *Journal for Research in Mathematics Education*, 24:442–459, 1993.
- [2] A. Birnbaum. Some latent trait models and their use in inferring an examinee’s ability. In F. M. Lord and M. R. Novick, editors, *Statistical Theories of Mental Test Scores*. Reading, MA: Addison-Wesley, 1968.
- [3] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [4] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 36(5):757–798, 2012.
- [5] K. R. Koedinger, S. D’Mello, E. A. McLaughlin, Z. A. Pardos, and C. P. Rosé. Data mining and education. *Wiley Interdisciplinary Reviews: Cognitive Science*, 6(4):333–353, 2015.
- [6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.
- [7] Z. Pardos, N. Heffernan, C. Ruiz, and J. Beck. The composition effect: Conjunctive or compensatory? an analysis of multi-skill math questions in its. In *Educational Data Mining 2008*, 2008.
- [8] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.
- [9] G. Rasch. *Probabilistic Models for Some Intelligence and Attainment Tests*. Danmarks Paedagogiske Institut, Copenhagen, 1960.
- [10] K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20:345–354, 1983.
- [11] W. J. van der Linden and R. K. Hambleton. *Handbook of Modern Item Response Theory*. Springer Science & Business Media, 2013.
- [12] S. E. Whitely. Multicomponent latent trait models for ability tests. *Psychometrika*, 45(4):479–494, 1980.
- [13] M. Wilson and P. De Boeck. Descriptive and explanatory item response models. In P. De Boeck and M. Wilson, editors, *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach*. New York: Springer-Verlag, 2004.
- [14] B. D. Wright and J. M. Linacre. Dichotomous rasch model derived from specific objectivity. *Rasch Measurement Transactions*, 1:5–6, 1987.

Sparse Neural Attentive Knowledge-based Models for Grade Prediction

Sara Morsy
Department of Computer Science
& Engineering
University of Minnesota
morsy@cs.umn.edu

George Karypis
Department of Computer Science
& Engineering
University of Minnesota
karypis@cs.umn.edu

ABSTRACT

Grade prediction for future courses not yet taken by students is important as it can help them and their advisers during the process of course selection as well as for designing personalized degree plans and modifying them based on their performance. One of the successful approaches for accurately predicting a student's grades in future courses is Cumulative Knowledge-based Regression Models (CKRM). CKRM learns shallow linear models that predict a student's grades as the similarity between his/her knowledge state and the target course. A student's knowledge state is built by linearly accumulating the learned provided knowledge components of the courses he/she has taken in the past, weighted by his/her grades in them. However, not all the prior courses contribute equally to the target course. In this paper, we propose a novel Neural Attentive Knowledge-based model (NAK) that learns the importance of each historical course in predicting the grade of a target course. Compared to CKRM and other competing approaches, our experiments on a large real-world dataset consisting of ~ 1.5 grades show the effectiveness of the proposed NAK model in accurately predicting the students' grades. Moreover, the attention weights learned by the model can be helpful in better designing their degree plans.

Keywords

grade prediction, knowledge-based models, neural networks, attention networks, undergraduate education

1. INTRODUCTION

The average six-year graduation rate across four-year higher-education institutions has been around 59% over the past 15 years [9, 2], while less than half of college graduates finish within four years [2]. These statistics pose challenges in terms of workforce development, economic activity and national productivity. This has resulted in a critical need for analyzing the available data about past students in order to provide actionable insights to improve college student

graduation and retention rates.

One approach for improving graduation and retention rates is to help students make more informed decisions about selecting the courses they register for in each term, such that the knowledge they have acquired in the past would prepare them to succeed in the next-term enrolled courses. Polyzou *et al.* [15] proposed course-specific linear models that learn the importance (or weight) of each previously-taken term towards accurately predicting the grade in a future course. One limitation of this approach is that in order to make accurate predictions, the model needs to have sufficient training data for each (prior, target) pair. Morsy *et al.* [13] developed Cumulative Knowledge-based Regression Models (CKRM) that also build on the idea of accumulating knowledge over time. CKRM predicts a student's grades as the similarity between his/her knowledge state and the target course. Both a student's knowledge state and a target course are represented as low-dimensional embedding vectors and the similarity between them is modeled by their inner product. A student's knowledge state is implicitly computed as a linear combination of the so-called provided knowledge component vectors of the previously-taken courses, weighted by his/her grades in them. Though CKRM was shown to provide state-of-the-art grade prediction accuracy, it is limited in that it assumes that all historical courses contribute equally in estimating the student's grade in a future course. Intuitively, students take courses from different departments, and each course would require an acquisition of knowledge from a few other courses, with different weights.

Motivated by the success of neural attentive networks in different fields [7, 12, 6, 1, 20], in this paper, we improve upon CKRM by learning the different importance of previously-taken courses in estimating the grade of a future course. We leverage the recent advances in neural attentive networks to learn these different weights, by employing both softmax and sparsemax activation functions that output posterior probabilities, i.e., attention weights, for the prior courses. The sparsemax function has an additional benefit of truncating the small probability values to zero, assigning zero effect to the irrelevant prior courses when predicting a target course's grade.

The main contributions of this work are as follows:

1. We propose a Neural Attentive Knowledge-based model

Sara Morsy and George Karypis "Sparse Neural Attentive Knowledge-based Model for Grade Prediction" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 366 - 371

(NAK) for grade prediction that improves upon CKRM by employing the attention mechanism in neural networks to learn the different importance of the prior courses towards predicting the grades of target courses. To our knowledge, this is the first work to apply attentive neural networks to grade prediction.

2. We leverage the recent sparsemax activation function for the attention mechanism that produces sparse attention weights instead of soft attention weights.
3. We performed an extensive experimental evaluation on a real world dataset obtained from a large university that spans a period of 16 years and consists of ~ 1.5 grades. The results show that our proposed NAK model significantly improves the prediction accuracy compared to the competing models. In addition, the results show the effectiveness of the attention mechanism in learning the different importance of the previously-taken courses towards each target course, which can help in designing better degree plans and more informed course selection decisions.

2. DEFINITIONS AND NOTATIONS

Boldface uppercase and lowercase letters will be used to represent matrices and vectors, respectively, e.g., \mathbf{G} and \mathbf{p} . The i th row of matrix \mathbf{P} is represented as \mathbf{p}_i^T , and its j th column is represented as \mathbf{p}_j . The entry in the i th row and j th column of matrix \mathbf{G} is denoted as $g_{i,j}$. A predicted value is denoted by having a hat over it (e.g., \hat{g}).

Matrix \mathbf{G} will represent the $m \times n$ student-course grades matrix, where $g_{s,c}$ denotes the grade that student s obtained in course c , relative to his/her average previous grade. Following the row-centering technique that was first proposed by Polyzou *et al.* [15], we subtract each student's grade from his/her average previous grade, since this was shown to significantly improve the prediction accuracy of different models. As there can be some students who achieved the same grades in all their prior courses, and hence their relative grades will be zero, in this case, we assigned a small value instead, i.e., 0.01. This is to prevent a prior course from not being considered in the model computation. A student s enrolls in sets of courses in consecutive terms, numbered relative to s from 1 to the number of terms in he/she has enrolled in the dataset. A set $\mathcal{T}_{s,w}$ will denote the set of courses taken by student s in term w .

3. RELATED WORK

3.1 Grade Prediction Methods

Grade prediction approaches for courses not yet taken by students have been extensively explored in the literature [16, 17, 8, 18, 15, 13, 5]. In this section, we review some research in grade prediction that is most relevant to our work.

3.1.1 Course-Specific Regression Models (CSR)

A more recent and natural way to model the grade prediction problem is to model the way the academic degree programs are structured. Each degree program would require the student to take courses in a specific sequencing such that the knowledge acquired in previous courses are required for the student to perform well in future courses. Polyzou *et al.* [15] developed course-specific linear regression

models (CSR) that build on this idea. A student's grade in a course is estimated as a linear combination of his/her grades in previously-taken courses, with different weights learned for each (prior, target) course pair. For a student s and a target course j , the predicted grade is estimated as:

$$\hat{g}_{s,j} = cb_j + \sum_{i \in \mathcal{P}} w_{i,j} g_{s,i}, \quad (1)$$

where cb_j is the bias terms for course j , $w_{i,j}$ is the weight of course i towards predicting the grade of course j , $g_{s,i}$ is the grade of student s in course i , and \mathcal{P} is the set of courses taken by s prior to taking course j . To achieve high prediction accuracy, CSR requires sufficient training data for each (prior, target) pair, which can hinder these models from good generalization.

3.1.2 Cumulative Knowledge-based Regression Models (CKRM)

Morsy *et al.* [13] developed Cumulative Knowledge-based Regression Models (CKRM), which is also based on the fact that the student's performance in a future course is based on his/her performance in the previously-taken courses. It assumes that a space of knowledge components exists such that each course provides a subset of these components as well as requires the knowledge of some of these components from the student in order to perform well in it. The student by taking a course thus acquires its knowledge components in a way that depends on his/her grade in that course. The overall knowledge acquired by the student after taking a set of courses is then represented by a knowledge state vector that is computed as the sum of the knowledge component vectors of those courses, weighted by his/her grades in them. Let \mathbf{p}_i denote the provided knowledge component vector for course i . The knowledge state vector for student s at term t can be expressed as follows:

$$\mathbf{k}_{s,t} = \sum_{w=1}^{t-1} \xi(s, w, t) \sum_{i \in \mathcal{T}_{s,w}} \left(g_{s,i} \mathbf{p}_i \right), \quad (2)$$

where $g_{s,i}$ is the grade that student s obtained on course i , and $\xi(s, w, t)$ is a time-based exponential decaying function designed to de-emphasize courses that were taken a long time ago.

Given the student's knowledge state vector prior to taking a course and that course's required knowledge component vector, denoted as \mathbf{r}_j , CKRM estimates the student's expected grade in that course as the inner product of these two vectors, i.e.,

$$\hat{g}_{s,j} = cb_j + \mathbf{k}_{s,t}^T \mathbf{r}_j, \quad (3)$$

where cb_j is as defined in Eq. 1, and $\mathbf{k}_{s,t}$ is the corresponding knowledge state vector. These course-specific linear models are estimated from the historical grade data and can be considered as capturing and weighting the knowledge components that a student needs to have accumulated in order to perform well in a course.

3.2 Neural Attentive Models

Our work relies on the attention mechanism, which has been recently introduced in neural networks and was shown to improve the performance of different models and give better

explanations to the importance of different objects towards a target object [6, 20, 7, 3]. Our work leverages several advances in this area. The most commonly-used activation function for the attention mechanism is the softmax function, which is easily differentiable and gives soft posterior probabilities that normalize to 1. A major disadvantage of the softmax function is that it assumes that each object contributes to the compressed representation, which may not always hold in some domains. To solve this, we need to output sparse posterior probabilities and assign zero to the irrelevant objects. Martins *et al.* [11] proposed the sparsemax activation function, which has the benefit of assigning zero probabilities to some output variables that may not be relevant for making a decision. This is done by defining a threshold, below which small probability values are truncated to zero. We also leverage the controllable sparsemax activation function recently proposed by Laha *et al.* [10] that controls the desired degree of sparsity in the output probabilities. This is done by adding an L2 regularization term that is to be maximized in the loss function. This will potentially encourage larger probability values for some objects, moving the rest to zero.

4. PROPOSED MODEL

4.1 Motivation

Consider a sample student who is declared in a Computer Science major and is in his/her second or third year in college. Table 1 shows the set of prior courses that this student has already take and the set of courses that this student is planning on taking the next term. With CKRM (Section 3.1.2), all these prior courses would contribute equally to predicting the grade of each target course. However, we can see that, intuitively, from the courses' names, there are courses that are strongly related to each target course and other courses that are irrelevant to it. For instance, it is reasonable to expect that the Intermediate German II course is more related to the Intermediate German I course than any of the other courses that the student has already taken. Along the same lines, we expect that the Algorithms and Data Structures course is more related to other Computer Science courses, such as the Advanced Programming Principles and the Program Design and Development courses. Assuming equal contribution among these prior courses can hinder the grade prediction model from accurately learning the course representations, and hence lead to poor predictions.

4.2 Overview

In this work, we present our Neural Attentive Knowledge-based model, NAK, which predicts a students' grades in future courses by employing an attention mechanism on the prior courses. We use CKRM as the underlying model (see Section 3.1.2).

4.3 Attention-based Pooling Layer for Prior Courses

In order to learn the different contributions of the prior courses in estimating the student's grade in a future course, we can employ the CSR technique (see Section 3.1.1) that learns the importance of each prior course in estimating the grade of each future course. Thus, we would estimate a

knowledge state vector for each target course j , using the following equation:

$$\mathbf{k}_{s,t,j} = \sum_{w=1}^{t-1} \sum_{i \in \mathcal{T}_w} \left(a_{i,j} g_{s,i} \mathbf{p}_i \right), \quad (4)$$

where $a_{i,j}$ is a learnable parameter that denotes the attention weight of course i in contributing to student s 's knowledge state when predicting s 's grade in course j . Note that we have removed the time-decaying function $\xi(s, w, t)$ that was used in CKRM (see Eq. 2), since it would be implicitly included in the attention weights. However, this solution requires sufficient training data for each (i, j) pair in order to be considered an accurate estimation.

In order to be able to have accurate attention weights between all pairs of prior and target courses, even the ones that do not appear together in the training data, we propose to use the attention mechanism that was recently used in neural networks [1, 19]. The main idea is to estimate the attention weight $a_{i,j}$ from the embedding vectors for courses i and j .

In order to compute the similarity between the embeddings of prior course i and target course j , we use a single-layer perceptron as follows:

$$z_{i,j} = \mathbf{h}^T \text{RELU}(\mathbf{W}(\mathbf{q}_i \odot \mathbf{r}_j) + \mathbf{b}), \quad (5)$$

where $\mathbf{q}_i = g_{s,i} \mathbf{p}_i$ denotes the embedding of the prior course i , weighted by the student's grade in it, \odot denotes the Hadamard product, and $\mathbf{W} \in \mathcal{R}^{l \times d}$ and $\mathbf{b} \in \mathcal{R}^l$ denote the weight matrix and bias vector that project the input into a hidden layer, respectively, and $\mathbf{h} \in \mathcal{R}^l$ is a vector that projects the hidden layer into an output attention weight, where d and l denote the number of dimensions of the embedding vectors and attention network, respectively. RELU denotes the Rectified Linear Unit activation function that is usually used in neural attentive networks.

4.3.1 Softmax Activation Function

The most common activation function used for computing these attention weights is the softmax function [19]. Given a vector of real weights \mathbf{z} , the softmax activation function converts it to a probability distribution, which is computed component-wise as follows:

$$\text{softmax}_i(\mathbf{z}) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}. \quad (6)$$

We will refer to the NAK model that uses the softmax activation function as **NAK(soft)**.

4.3.2 Sparsemax Activation Function

Although the softmax activation function has been used to design attention mechanisms in many domains [14, 1, 6, 12, 7], we believe that using it for grade prediction is not optimal. Since a student enrolls in several courses, and each course requires knowledge from one or a few other courses, we hypothesize that some of the prior courses should have no effect, i.e., zero attention, towards predicting a target course's grade. We thus leverage a recent advance, the sparsemax activation function [11], to learn sparse attention weights. The idea is to define a threshold, below

Table 1: Sample of prior and target courses for a Computer Science student at University X.

Prior Courses	Target Course
Calculus I, Beginning German, Operating Systems, Intermediate German I, University Writing, Introductory Physics, Peotics in Film, Program Design & Development, Philosophy, Linear Algebra, Internet Programming, Stone Tools to Steam Engines, Advanced Programming Principles, Computer Networks	Intermediate German II
	Probability & Statistics
	Algorithms & Data Structures

which small probability values are truncated to zero. Let $\Delta^{K-1} := \{\mathbf{x} \in \mathbb{R}^K | \mathbf{1}^T \mathbf{x} = 1, \mathbf{x} \geq \mathbf{0}\}$ be the $(K-1)$ -dimensional simplex. The sparsemax activation function tries to solve the following equation:

$$\text{sparsemax}(\mathbf{z}) = \underset{\mathbf{x} \in \Delta^{K-1}}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{z}\|^2, \quad (7)$$

which, in other words, returns the Euclidean projection of the input vector \mathbf{z} onto the probability simplex.

In order to obtain different degrees of sparsity in the attention weights, Laha *et al.* [10] developed a generic probability mapping function for the sparsemax activation function, which they called **sparsegen**, and is computed as follows:

$$\text{sparsegen}(\mathbf{z}; \gamma) = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{z}\|^2 - \gamma \|\mathbf{x}\|^2, \quad (8)$$

where $\gamma < 1$ controls the L2 regularization strength of \mathbf{x} . An equivalent formulation for sparsegen was formed as:

$$\text{sparsegen}(\mathbf{z}; \gamma) = \text{sparsemax}\left(\frac{\mathbf{z}}{1-\gamma}\right), \quad (9)$$

which, in other words, applies a temperature parameter to the original sparsemax function. Varying this temperature parameter can change the degree of sparsity in the output variables. By setting $\gamma = 0$, sparsegen becomes equivalent to sparsemax. We will refer to the NAK model that uses the sparsegen activation function as **NAK(sparse)**.

4.4 Prediction

NAK then predicts the grade for student s in course j that he/she takes at term t as:

$$\hat{g}_{s,j} = cb_j + \mathbf{k}_{s,t,j}^T \mathbf{r}_j. \quad (10)$$

4.5 Optimization

We use the mean squared error (MSE) loss function to estimate the parameters of NAK. We minimize the following regularized RMSE loss:

$$L = -\frac{1}{2N} \sum_{s,c \in \mathbf{G}} (g_{s,c} - \hat{g}_{s,c})^2 + \alpha \|\Theta\|^2, \quad (11)$$

where N is the number of grades in \mathbf{G} . The hyper-parameter α controls the strength of L2 regularization to prevent overfitting, and $\Theta = \{\{\mathbf{cb}\}, \{\mathbf{p}_i\}, \{\mathbf{r}_i\}, \mathbf{W}, \mathbf{b}, \mathbf{h}\}$ denotes all trainable parameters of NAK.

The optimization problem is solved using AdaGrad algorithm [4], which applies an adaptive learning rate for each parameter. It randomly draws mini-batches of a given size from the training data and updates the related model parameters. The source code can be found here: <https://urlzs.com/iH8G>.

5. EVALUATION METHODOLOGY

5.1 Dataset

The data used in our experiments was obtained from the University of Minnesota (UMN), which includes 96 majors from 10 different colleges, and spans the years 2002 to 2017. At the University, the letter grading system used is A–F, which is converted to the 4–0 scale using the standard letter grade to GPA conversion. We removed any grades that were taken as pass/fail. The final dataset includes $\sim 54,000$ students, 5,800 courses, and 1,450,000 grades in total.

5.2 Generating Training, Validation and Test Sets

At UMN, there are three terms, Fall, Summer and Spring. We used the data from 2002 to Spring 2015 (inclusive) as the training set, the data from Spring 2016 to Fall 2016 (inclusive) as the validation set, and the data from Summer 2016 to Summer 2017 (inclusive) as the test set. For a target course taken by a student to be predicted, that student must have taken at least four courses prior to the target course, in order to have sufficient data to compute the student's knowledge state vector. We excluded any courses that do not appear in the training set from the validation and test sets.

5.3 Comparison Methods

We compared the performance of our NAK model against the following grade prediction approaches:

1. **Matrix Factorization (MF):** This approach predicts the grade for student s in course i as:

$$\hat{g}_{s,i} = \mu + sb_s + cb_i + \mathbf{u}_s^T \mathbf{v}_i, \quad (12)$$

where μ , sb_s and cb_i are the global, student and course bias terms, respectively, and \mathbf{u}_s and \mathbf{v}_i are the student and course latent vectors, respectively. We used the squared loss function with L2 regularization to estimate this model.

2. **KRM(sum):** This is CKRM the method described in Section 3.1.2.
3. **KRM(avg):** This is similar to the KRM(sum) method, except that the prior courses' embeddings are aggregated with mean pooling instead of summation. It was shown in later studies, e.g. [17], that it performs better than KRM(sum).

We implemented KRM(sum) and KRM(avg) with a neural network architecture and optimization similar to that of NAK.

Table 2: Comparison between the baseline and proposed models.

Model	Parameters					RMSE	PTA0	PTA1	PTA2
MF	16	1E-04	1E-02	–	–	0.724	25.7	58.6	79.5
KRM(sum)	32	1E-07	7E-04	0.3	–	0.584	32.6	70.1	87.7
KRM(avg)	32	1E-07	7E-04	0.0	–	0.584	34.9	70.6	87.7
NAK(soft)	32	1E-07	7E-04	3	–	0.589 (–0.9%)	35.3† (1.1%)	71.8 (1.7%)	88.0† (0.3%)
NAK(sparse)	32	1E-07	7E-04	4	0.5	<u>0.574</u> †‡ (1.7%)	<u>35.3</u> † (1.1%)	<u>72.1</u> (2.1%)	<u>88.7</u> † (1.1%)

The Parameters columns denote the following model parameters that were selected: for MF, the parameters are: the number of latent dimensions, the L2 regularization parameter, and the learning rate; for KRM(sum) and KRM(avg), the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, and the time-decaying parameter λ ; for NAK, the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, and the number of latent dimensions for the MLP attention mechanism; and for NAK(sparse), the last parameter denotes the L2 regularization parameter γ for the sparsegen activation function. Underlined entries represent the best performance in each metric. The † and ‡ symbols are used to denote results that are statistically significant over the best performing baseline metric, and NAK(soft), respectively, using the Student’s paired t -test with a p -level < 0.1 . Numbers in parentheses denote the percentage of improvement over the best baseline value in each metric.

5.4 Model Selection

We performed an extensive search on the parameters of the proposed and baseline models to find the set of parameters that gives us the best performance for each model.

For all proposed and competing models, the following parameters were used. The number of latent dimensions for course embeddings was chosen from the set of values: {8, 16, 32}. The L2 regularization parameter was chosen from the values: {1e-5, 1e-7, 1e-3}. Finally, the learning rate was chosen from the values: {0.0007, 0.001, 0.003, 0.005, 0.007}. For the proposed NAK models, the number of latent dimensions for the MLP attention mechanism was selected in the range [1, 4]. For KRM(sum) and KRM(avg), the time-decaying parameter λ was chosen from the set of values: {0, 0.3, 0.5, 0.7, 1.0}.

The training set was used for estimating the models, whereas the validation set was used to select the best performing parameters in terms of the overall MSE of the validation set.

5.5 Evaluation Methodology and Metrics

The grading system used by the University uses a 12 letter grade system (i.e., A, A–, B+, ... F). We will refer to the difference between two successive letter grades (e.g., B+ vs B) as a *tick*. We converted the predicted grades into their closest letter grades. We assessed the performance of the different approaches based on the Root Mean Squared Error (RMSE) as well as how many ticks away the predicted grade is from the actual grade, which is referred to as *Percentage of Tick Accuracy*, or PTA. We computed the percentage of grades predicted with no error (zero tick), within one tick, and within two ticks, which will be referred to as PTA0, PTA1, and PTA2, respectively.

6. EXPERIMENTAL RESULTS

6.1 Performance of the Proposed Models

Table 2 shows the performance of our proposed models. Using the sparsegen activation function instead of the softmax activation function improves the prediction accuracy, with a statistically significant improvement. This shows that using the sparsegen activation function to output sparse attention weights for the prior courses achieves better prediction accuracy than producing soft probabilities for all of them. This is expected, since the student’s prior courses may not be all relevant to the target course, as illustrated in Table 1.

6.2 Performance against Competing Methods

Table 2 also shows the performance of the competing models. Among the baseline methods, both KRM(sum) and KRM(avg) outperform MF. KRM(avg) outperforms KRM(sum) in PTA0 and PTA1. Both NAK(soft) and NAK(sparse) outperform all baseline methods. Even though the RMSE results of NAK(soft) is worse than these of the KRM variants, it achieved $\sim 1\%$, $\sim 2\%$ and 0.5% more accurate predictions within no, one, and two tick errors, respectively. Among all baseline and proposed methods, our NAK(sparse) model outperforms all baseline methods significantly, with achieving $\sim 2\%$ lower RMSE, and $\sim 1\%$ more accurate predictions within two ticks than KRM(avg). This shows that using the attention-based pooling layer on the prior courses to accumulate them can better predict the grades of students in their future courses.

6.3 Qualitative Analysis on the Prior Courses Attention Weights

Recall the motivational example for the Computer Science student, discussed in Section 4.1. This student had a set of prior courses and three target courses that we would like to predict his/her grades in (See Table 1). Using KRM(sum) or KRM(avg), all the prior courses would contribute equally to the prediction of each target course. Using our proposed NAK(sparse) model, the attention weights for the prior courses with each target course are shown in Table 3¹.

We can see that, using the sparsegen activation function, only a few prior courses are selected with non-zero attention weights, which are the most relevant to each target course.

For the Intermediate German II course, we can see that the student’s grade in it is most affected by two courses: the Intermediate German I course, and the University Writing course. The Intermediate German I course is listed as a pre-requisite course for the Intermediate German II course. Though the University Writing course is not listed as a pre-requisite course, after further analysis, we found out that the Intermediate German II course requires process-writing essays and are considered part of the grading system. Though the German courses are not part of the student’s degree program, and are taken by a small percentage of Computer

¹These results were obtained by learning NAK models to estimate the actual grades and not the row-centered grades. Also, we used $\mathbf{q}_i = \mathbf{p}_i$ in Eq. 5. This allowed us to get more interpretable results.

Table 3: The attention weights of the prior courses with each target course learned by NAK(sparse) for the sample student from Table 1.

Prior Courses	Target Course
Intermediate German I: 0.6980 , University Writing: 0.3020	Intermediate German II
Calculus I: 0.4737 , Physics: 0.3794 , Program Design & Development: 0.0717 , Operating Systems: 0.0497 , Computer Networks: 0.0255	Probability & Statistics
Operating Systems: 0.2927 , Advanced Programming Principles: 0.2582 , Linear Algebra: 0.2313 , Physics: 0.2178	Algorithms & Data Structures

Prior courses are sorted in non-increasing order w.r.t. to their attention weights with each target courses for clarity purposes.

Science students, our NAK model was able to learn accurate attention weights for them.

The other two target courses, Probability and Statistics, and Algorithms and Data Structures, have totally different prior courses with the largest attention weights, which are more related to them.

These results illustrate that our proposed NAK model was able to uncover the listed as well as the hidden/informal pre-requisite courses without any supervision given to the model.

7. CONCLUSION

In this work, we presented a method to improve the grade prediction accuracy, by learning the weights of the prior courses towards predicting the grade of each target course. To this end, we employed the attention mechanism on the prior courses that learns the different contributions of these courses towards each target course. We employed both a softmax and a sparsemax activation function that produce soft and sparse attention weights, respectively. The proposed models are able to capture the listed as well as the hidden pre-requisite courses for the target courses, which can be better used to design better degree plans. Our experiments showed that our models significantly outperformed the competing methods, indicating the value of the attention mechanism on the prior courses.

Acknowledgement

This work was supported in part by NSF (1447788, 1704074, 1757916, 1834251), Army Research Office (W911NF1810344), Intel Corp, and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute, <http://www.msi.umn.edu>.

8. REFERENCES

- [1] D. Bahdanau and *et al.*. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [2] J. Braxton and *et al.*. *Understanding and Reducing College Student Departure: ASHE-ERIC Higher Education Report, Volume 30, Number 3*, volume 16. John Wiley & Sons, 2011.
- [3] J. Chen and *et al.*. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *SIGIR*. ACM, 2017.
- [4] J. Duchi and *et al.*. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12(Jul):2121–2159, 2011.
- [5] A. Elbadrawy and *et al.*. Domain-aware grade prediction and top-n course recommendation. In *RecSys*. ACM, 2016.
- [6] X. He and *et al.*. Neural factorization machines for sparse predictive analytics. In *SIGIR*. ACM, 2017.
- [7] X. He and *et al.*. Nais: Neural attentive item similarity model for recommendation. *TKDE*, 30(12):2354–2366, 2018.
- [8] Q. Hu and *et al.*. Course-specific markovian models for grade prediction. In *PAKDD*. Springer, 2018.
- [9] G. Kena and *et al.*. The condition of education 2016. nces 2016-144. *National Center for Education Statistics*, 2016.
- [10] A. e. Laha. On controllable sparse alternatives to softmax. In *NIPS*, pages 6423–6433, 2018.
- [11] A. e. Martins. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *ICML*, pages 1614–1623, 2016.
- [12] L. Mei and *et al.*. An attentive interaction network for context-aware recommendations. In *CIKM*. ACM, 2018.
- [13] S. Morsy and *et al.*. Cumulative knowledge-based regression models for next-term grade prediction. In *SDM*. SIAM, 2017.
- [14] A. Parikh and *et al.*. A decomposable attention model for natural language inference. In *EMNLP*, 2016.
- [15] A. Polyzou and *et al.*. Grade prediction with course and student specific models. In *PAKDD*. Springer, 2016.
- [16] Z. Ren and *et al.*. Grade prediction with temporal course-wise influence. In *EDM*, 2017.
- [17] Z. Ren and *et al.*. Ale: Additive latent effect models for grade prediction. In *SDM*. SIAM, 2018.
- [18] M. Sweeney and *et al.*. Next-term student performance prediction: A recommender systems approach. *EDM*, 8(1):22–51, 2016.
- [19] A. e. Vaswani. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.
- [20] J. Xiao and *et al.*. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *IJCAI*. AAAI Press, 2017.

Modeling person-specific development of math skills in continuous time: New evidence for mutualism

Lu Ou
ACTNext by ACT, Inc.
500 ACT Drive
Iowa City, IA 52243
lu.ou@act.org

Timo Bechger
ACTNext by ACT, Inc.
500 ACT Drive
Iowa City, IA 52243
timo.bechger@act.org

Abe D. Hofman
University of Amsterdam
Nieuwe Achtergracht 129-B
1018 WS Amsterdam
The Netherlands
a.d.hofman@uva.nl

Gunter Maris
ACTNext by ACT, Inc.
500 ACT Drive
Iowa City, IA 52243
gunter.maris@act.org

Vanessa R. Simmering
ACTNext by ACT, Inc.
500 ACT Drive
Iowa City, IA 52243
vanessa.simmering@act.org

Han L. J. van der Maas
University of Amsterdam
Nieuwe Achtergracht 129-B
1018 WS Amsterdam
The Netherlands
h.l.j.vandermaas@uva.nl

ABSTRACT

In this study, we fitted a mixed-effects nonlinear continuous-time mutualism model of skill development proposed by van der Maas et al. (2006) to naturally collected irregularly spaced time series data from an online adaptive practice system for mathematics called Math Garden. Results showed that the mutualism model provided a better fit to the data than a g -factor model. The paper illustrates continuous-time modeling of irregularly-spaced multivariate time series data that are increasingly prevalent in modern learning systems.

Keywords

mutualism model, continuous-time model, mixed-effects model

1. INTRODUCTION

For the past century, generations of researchers have continued to pursue explanations for the consistent positive correlations between diverse sets of cognitive ability tests, known as *the positive manifold* [25, 29]. Heated debates went on about whether there is a potential biologically based g -factor that causes the development of general intelligence as well as the positive manifold [26, 7, 27, 11, 9]. Although researchers have not reached consensus, there is a shift from conceptualizing cognitive development as merely reflective, as in factor analysis, to thinking of it as formative [2, 14, 27]. In a formative model, the positive manifold is an emergent property that results from within-person changes and connections over time. This ontological stance implies that research needs to focus on understanding the causal relationships that underlie cognitive development to guide effective efforts to predict and intervene in students' learning.

Lu Ou, Abe Hofman, Vanessa Simmering, Timo Bechger, Gunter Maris and Han van der Maas "Modeling person-specific development of math skills in continuous time: New evidence for mutualism" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 372 - 377

Various mathematical representations encompassing continuous and discrete variables have been proposed to describe the mechanistic changes and sources of individual differences in cognition [28, 9, 32, 21]. From a developmental perspective, cognitive abilities develop as a dynamic system with reciprocal interactions between the elements of the system causing the developmental pathways of each of the elements [29]. In the *Mutualism* model of intelligence, elements of a system interact with each other in a collaborative way to achieve mutual benefits. This provides an alternative explanation for the positive manifold, other than the g -factor approach, and only requires sparse, weak, and even some negative interactions to produce positive correlations [28].

In the current study, we take advantage of massive time series data collected with an online learning environment for mathematics [12] and propose a method to fit the mutualism model to this dataset. We aim to examine potential reciprocal interactions of mathematical skills in two domains — counting and addition — in children's learning and practicing mathematics online. We build a model that takes into account individual differences in the learning processes by allowing individuals to start in different positions and by including random effects in key parameters of an otherwise group-based mutualism model. Note that this is the first application of the nonlinear mutualism differential equation model to empirical data, providing an evaluation of how well the theoretical account proposed by van der Maas et al. [29] can capture changes in children's mathematical skill development over time. In addition, we pioneer the use of continuous-time models to analyze irregularly-spaced data that arise when students use educational technology in realistic settings, and show that the estimation framework implemented in the *dynr* R package [19, 20] can handle nonlinear equations and mixed effects that explain both between-person and within-person differences.

In summary, the contribution of the work is three-fold: 1) providing new evidence of reciprocal interactions in mathematics skill development as a pioneer in fitting the nonlinear mutualism model to empirical data; 2) presenting a way to

analyze the irregularly spaced multivariate time series data commonly seen in learning systems; and 3) demonstrating the use of state-space approaches in estimating parameters of mixed-effects dynamic models.

In the following sections, we first explain the mathematical model we use to characterize the learning processes, and the estimation procedure. We then present how the empirical data in the current study were collected, and the sample characteristics. The paper ends with a discussion of the results and their implications for education.

2. THE MUTUALISM MODEL

In biology, the term *Mutualism* is used for a relation between species populations where different species organically interact with each other to maintain sustainable growth [1]. Biologists routinely use the Lotka-Volterra model [16, 30] to study the dynamics of such relations, which inspired van der Maas and colleagues [29] to propose the same model, referred to as the mutualism model, to study the dynamics of cognitive development, where elements of a cognitive system interact with each other to achieve mutual benefit.

2.1 The Lotka-Volterra Model

Mathematically, the mutualism model can be expressed using generalized N-subject Lotka-Volterra equations as

$$\begin{aligned} d\mathbf{x}(t) &= \mathbf{F}(\mathbf{x}_1(t), \mathbf{x}_2(t), \dots, \mathbf{x}_N(t))dt \\ &= \left[\rho_i x_i(t) \left(1 - \frac{x_i(t) + \sum_{j \neq i} a_{ij} x_j(t)}{K_i} \right) \right] dt, \end{aligned} \quad (1) \quad (2)$$

where $i, j = 1, 2, \dots, N$ indicates different elements of a dynamic system, and t is continuous time. Here, the elements are the counting and addition skills. The differential of vector $\mathbf{x}(t)$ with respect to t denotes the change in $\mathbf{x}(t)$ within an infinitely small time interval.

The model assumes logistic growth. The ρ_i are growth parameters that determine the steepness of the logistic growth function associated with each $x_i(t)$, and the K_i are the carrying capacity parameters that represent the limited resources in the system, such as limited attention and working memory one can allocate in learning. The a_{ij} are interaction parameters that specify the relations between each pair of x_i and x_j in development. With all $a_{ij} = 0$, the change of the latent variable $x_i(t)$ follows a simple logistic curve that converges to an equilibrium state of K_i , regardless of its starting position. The system is collaborative if the Jacobian matrix $\frac{\partial \mathbf{F}}{\partial \mathbf{x}}$ is positive definite, and is competitive otherwise. If, for all i , $x_i(t)$ and ρ_i only take positive values, then it is possible to show that as long as the combined consumption of resources $x_i(t) + \sum_{j \neq i} a_{ij} x_j(t)$ does not exceed the carrying capacity K_i , $x_i(t)$

will continue to increase to its equilibrium. Further, when the interaction parameters a_{ij} are negative (or $-a_{ij}$ are positive) for all $j \neq i$, $x_i(t)$ can develop even beyond the original carrying capacity K_i , as a benefit of the collaboration with the other processes. On the other hand, when the parameters $a_{ij}, j \neq i$ are positive, $x_i(t)$ can never reach the full potential K_i , as a loss due to competition. van der Maas and colleagues [29] showed that when $-a_{i,j}$ is positive and

less than 1, the mutualism model can result in the positive manifold.

2.2 State-space Representation

If we take into account individual differences in the mutualism model, as well as process noise and measurement errors¹ that may occur alongside the manifestation of the mutualism process, we obtain a state-space representation of the mutualism model:

$$d\mathbf{x}_s(t) = \mathbf{F}_s(\mathbf{x}_s(t))dt + d\mathbf{w}_s(t) \quad (3)$$

$$\mathbf{F}_s(\mathbf{x}_s(t)) = \begin{bmatrix} \rho_1 x_{1,s}(t) \left(1 - \frac{x_{1,s}(t) + a_{12} x_{2,s}(t)}{K_1 + b_{1,s}} \right) \\ \rho_2 x_{2,s}(t) \left(1 - \frac{x_{2,s}(t) + a_{21} x_{1,s}(t)}{K_2 + b_{2,s}} \right) \end{bmatrix} \quad (4)$$

$$\mathbf{y}_s(t_{s,k}) = \mathbf{x}_s(t_{s,k}) + \boldsymbol{\epsilon}_s(t_{s,k}), \quad (5)$$

$$\boldsymbol{\epsilon}_s(t_{s,k}) \sim N \left(\mathbf{0}, \boldsymbol{\Sigma}_\epsilon = \begin{bmatrix} \sigma_{\epsilon,1}^2 & 0 \\ 0 & \sigma_{\epsilon,2}^2 \end{bmatrix} \right), \quad (6)$$

where the subscript s indexes individuals, and $k = 1, 2, \dots, T_s$ indexes the k th discrete person-specific measurement occasions $t_{s,k}$. The vector $\mathbf{x}_s(t)$ contains the latent counting and addition skills $x_{1,s}(t)$ and $x_{2,s}(t)$ for an individual s , manifested as $\mathbf{y}_s(t_{s,k})$ in a measurement model with serially independent Gaussian measurement errors $\boldsymbol{\epsilon}_s(t_{s,k})$. The differential of $\mathbf{x}_s(t)$ is determined by the systematic dynamic functions $\mathbf{F}_s(\cdot)$ and the differential of process noise $\mathbf{w}_s(t)$ that follows a Wiener process (i.e., a continuous-time version of random walk, [10]), with a diffusion matrix $\mathbf{Q} = \begin{bmatrix} \sigma_{w,1}^2 & 0 \\ 0 & \sigma_{w,2}^2 \end{bmatrix}$. Person-specific random effects $\begin{bmatrix} b_{1,s} \\ b_{2,s} \end{bmatrix}$ are added to the carrying capacity parameters, and are assumed to follow a normal distribution with mean $\mathbf{0}$ and a covariance matrix of $\boldsymbol{\Sigma}_b = \begin{bmatrix} \sigma_{b,11}^2 & \sigma_{b,12}^2 \\ \sigma_{b,12}^2 & \sigma_{b,22}^2 \end{bmatrix}$.

The initial condition, or the distribution of the variables at the first available time point, of the dynamic process $\mathbf{x}_s(t_{s,1})$ is assumed to follow a multivariate normal distribution with mean $\begin{bmatrix} \mu_{1,1} \\ \mu_{1,2} \end{bmatrix}$ and variance $\begin{bmatrix} \sigma_{1,11}^2 & \sigma_{1,12}^2 \\ \sigma_{1,12}^2 & \sigma_{1,22}^2 \end{bmatrix}$.

2.3 An Alternative G-factor Model

In order to explore the fit of the mutualism model to empirical data compared to the g -theory, a comparable state-space

¹Process noise is distinct from measurement error in that the former is associated with random behavior in the underlying process, whereas the latter depends on the measurement process, device, and other environmental influences that may affect the accuracy of the measurements. In an educational context, a correct guess without knowing an item can be seen as a measurement error, while a child having a good or bad day could contribute to the process noise. Whereas measurement error does not influence growth at the next time point, the process noise does steer the dynamical system.

one-factor model without interactions can be developed as

$$dx_s(t) = \rho_1 x_s(t) \left(1 - \frac{x_s(t)}{K_1 + b_{1,s}}\right) + dw_s(t) \quad (7)$$

$$x_s(t_{s,1}) \sim N(\mu_{1,1}, \sigma_{1,11}^2), \mathbf{Q} = [\sigma_{w,1}^2], \mathbf{\Sigma}_b = [\sigma_{b,11}^2]$$

$$\mathbf{y}_s(t_{s,k}) = \begin{bmatrix} 1 \\ \lambda \end{bmatrix} x_s(t_{s,k}) + \boldsymbol{\epsilon}_s(t_{s,k}), \quad (8)$$

$$\boldsymbol{\epsilon}_s(t_{s,k}) \sim N\left(\mathbf{0}, \mathbf{\Sigma}_\epsilon = \begin{bmatrix} \sigma_{\epsilon,1}^2 & 0 \\ 0 & \sigma_{\epsilon,2}^2 \end{bmatrix}\right),$$

where the observed variables are linearly linked with the single latent variable through a loading matrix of $\begin{bmatrix} 1 & \lambda \end{bmatrix}^\top$.

3. ESTIMATION

To estimate the random effects in the models, we augmented the latent variables $\mathbf{x}_s(t)$ with random effects \mathbf{b}_s to yield a new latent variable vector, $\mathbf{x}_s^*(t) = [\mathbf{x}_s(t) \ \mathbf{b}_s]^\top$. We then modified the differential equations, the measurement model, and the initial condition to incorporate this change of $\mathbf{x}_s^*(t)$.

We used the dynr R package [19, 20] to estimate the parameters in the mixed-effects mutualism model, as well as the baseline g -factor model, by numerically optimizing an approximate log-likelihood function obtained as a by-product of the continuous-discrete extended Kalman filter [15]. Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) were constructed to compare models. Details of the estimation algorithms can be found in [4, 20].

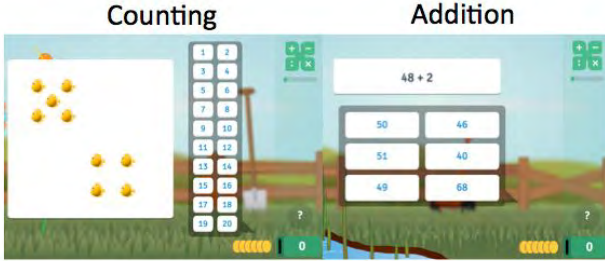


Figure 1: Screen shots of the counting and addition games in the Math Garden. Children give responses by clicking an option. The coins at the bottom disappear one per second, and reflect the scoring rule based on accuracy and response time.

4. EMPIRICAL STUDY

Here, we describe an application of the mutualism model.

4.1 Math Garden

We sampled data using a popular Dutch online adaptive practice and monitoring system called Math Garden [13]. The system consists of games that measure different mathematical skills, including counting and addition, as players practice their arithmetic skills through answering items. Figure 1 shows screen shots of two example items.

The system applies an explicit scoring rule for both speed and accuracy [17], visible to players as the number of coins they collect. For each item, a limit number of coins can be collected, and the number decreases by one at each additional second used to come up with the answer. In case of

a correct answer, the score equals the remaining time. If the answer is incorrect, the score is the negative remaining time. The scoring rule takes speed-accuracy trade-off into account, penalizes quick but incorrect answers, and encourages thoughtful responses.

Skill rating and item difficulty are estimated on-the-fly using the Elo-algorithm [6] which was originally developed for chess competitions between two players, and now has been adapted for pairing a player with an item [13]. The skill and difficulty estimates for a player and an item are updated at each “match” they are involved in, depending on the weighted difference between observed and expected correctness, the latter of which is entailed by the measurement model [17]. Evidence has shown high validity and reliability of the skill and difficulty estimates [13].

In the current study, our observed data are the continuous end-of-day skill ratings in different domains, rather than binary correctness for each item. Comparisons of Math Garden’s underlying measurement model [17] and the adapted Elo-algorithm [13] to other common models for binary responses in educational data mining — the Rasch model [23], additive factor models [3], performance factor analysis [22], and Bayesian knowledge tracing models [5] — are worth exploration, but beyond the scope of this paper.

4.2 Data Description

We selected a sample of children in grades 3–6, between the ages of 6 and 10 years old, who practiced counting and addition skills during at least 4 different months in the school year from September, 2016 to July, 2017, and had played at least 20 different days in each domain, with a minimum of 10 items per day. We excluded children whose parents indicated unwillingness to participate in Math Garden-related scientific research that was approved by the Ethics Committee of the psychology department of University of Amsterdam.

The resulting sample included a total of 2485 children, 51.07% male. The average age at which a child started to use Math Garden for practicing counting and addition during the school year was 7.23 years old ($SD = 1.03$). The original skill ratings could be negative, so we shifted them to the positive range by respectively adding 20 and 25 to the counting and addition scales. The over-time ebb-and-flow and variability of the skill ratings remain the same. From the second-by-second time stamps of the data points, we constructed continuous measures of time where each unit represents a week. Figure 2 shows the shifted skill ratings for three randomly selected individuals over time.

Distributions of the initial and ending skill ratings are plotted in Figure 3. At the first available time point for each individual, the counting skill ratings for all sampled children had a mean of 13.84 and a variance of 1.51, whereas the addition skill ratings had mean 12.94 and a larger variance of 10.56. At the last available time point for each individual, the ending counting estimates had a mean of 14.83 and variance of 1.49, while the ending addition estimates had a mean of 15.92 and variance of 9.05. Generally speaking, during the school year, more development is observed in the addition skill compared to the counting skill. There was more variability in children’s initial and ending addition skill ratings

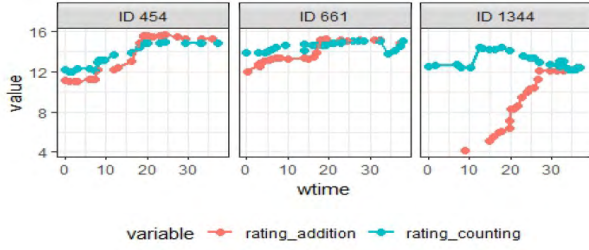


Figure 2: Individual time series data of counting and addition skill ratings for three children.

than in the counting domain.

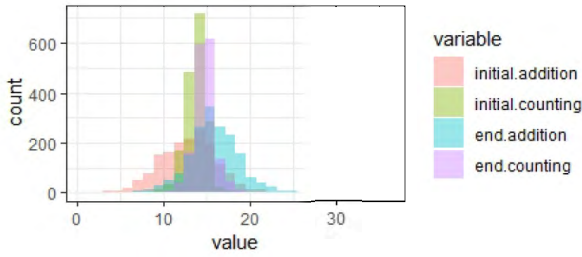


Figure 3: Histograms of the initial and ending skill ratings

The correlation between the initial addition and counting skill ratings was .73, and the correlation between the ending ratings was .79, confirming the positive manifold. Figure 4 shows the boxplot of within-person correlations between the skill ratings in the two domains. The mean of the within-person correlations was .55 ($SD = 0.38$). However, some negative values were observed at the significance level of 0.05 of the asymptotic p-values computed by the Hmisc R package [8]. For example, in Figure 2 the child with identification number 1344 had upward growth in the addition skill ratings and downward decline in the counting skill ratings. The downward decline may be due to an unexpected bump in the skill ratings that was higher than the child's equilibrium and hence resulted in a return to the equilibrium. Another possible explanation would be that, the child learned and practiced counting at school before addition, but forgetting took place as the child started to learn addition and practiced counting less. In such a case, there was a competition for attention and learning time between the skills in different domains, instead of a collaboration. Either case can be captured by the mutualism model.

The length of the individual time series of the counting skill ratings ranged from 20 to 177 days with a median of 29 days, and that of the addition skill ratings ranged from 20 to 192 days with a median of 42 days. The length of the interval between two time points represents the inactivity gap between two practice days of an individual for a mathematical skill, and ranged from 1 day to 18.29 weeks. The minimal gap length of a single time series had a median of 1 day across the sample in both domains, whereas the median maximum gap length was 4.86 weeks in the counting domain and 4 weeks in the addition domain. In Figure 5, the data of

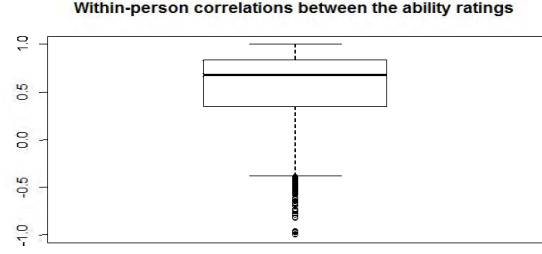


Figure 4: A boxplot of the within-person correlations of the addition and counting skill ratings.

ten randomly selected individuals illustrate the irregularly-spaced measurement occasions, as well as the unbalanced practices in each domain on a single day and across time. The mutualism model assumes a continuous integrative process of change even though we do not have measurements of each skill at all times.

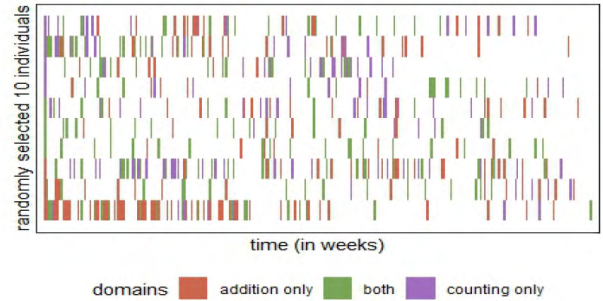


Figure 5: An illustration of the irregularly spaced time intervals of ten randomly selected individuals. Different colors represent the domains that an individual practiced during a specific day.

4.3 Empirical Results

The parameter estimates and model fit indices of both the mutualism model and the g -factor model were summarized in Table 1. All parameters were estimated to be significantly different from zero ($p < .05$). The estimates of the initial condition parameters ($\mu_{1,1}$, $\mu_{1,2}$, $\sigma_{1,1}^2$, $\sigma_{1,12}^2$, and $\sigma_{1,22}^2$) in both models were consistent with the sample mean and variance of the initial states. With lower AIC and BIC values, the mutualism model provided a better fit to the data compared to the g -factor model. Figure 6 shows the fit of the mutualism model to the observed data of four randomly selected individuals. The fitted trajectories were able to capture the changes of the observed paths for the individuals in both domains, suggesting a decent fit of the model to the data. In the mutualism model, the steepness parameters ρ_1 and ρ_2 were estimated to be close to zero, indicating that the overall development in skills was small and slow. The group-level equilibrium states K_1 and K_2 , for when there was no interaction between the processes, were estimated to about 10, but individual differences captured by the random effects b_1 and b_2 contribute to an estimated co-variance of $\begin{bmatrix} 1.04 & 0.09 \\ 0.09 & 1.06 \end{bmatrix}$. Estimates of the interaction parameters a_{12} and a_{21} were found to be significantly negative, so the inter-

Table 1: Parameter estimates (standard errors) and model fit indices

	Mutualism Model	g -factor Model
ρ_1	0.08 (0.002)	0.02 (0.001)
ρ_2	0.09 (0.001)	
a_{12}	-0.48 (0.005)	
a_{21}	-0.58 (0.004)	
K_1	10.04 (0.005)	15.98 (0.124)
K_2	10.05 (0.004)	
$\sigma_{w,1}^2$	0.34 (0.002)	0.10 (0.001)
$\sigma_{w,2}^2$	0.42 (0.001)	
$\sigma_{\epsilon,1}^2$	0.29 (0.001)	4.92 (0.025)
$\sigma_{\epsilon,2}^2$	0.34 (0.002)	0.08 (0.001)
$\mu_{1,1}$	13.85 (0.006)	13.87 (0.013)
$\mu_{1,2}$	12.92 (0.001)	
$\sigma_{1,11}^2$	1.67 (0.014)	1.74 (0.068)
$\sigma_{1,12}^2$	1.34 (0.045)	
$\sigma_{1,22}^2$	10.55 (0.056)	
$\sigma_{b,11}^2$	1.04 (0.005)	1.71 (0.310)
$\sigma_{b,12}^2$	0.09 (0.004)	
$\sigma_{b,22}^2$	1.06 (0.005)	
λ		1.02 (0.001)
AIC	388981.81	472908.58
BIC	389155.46	472995.41

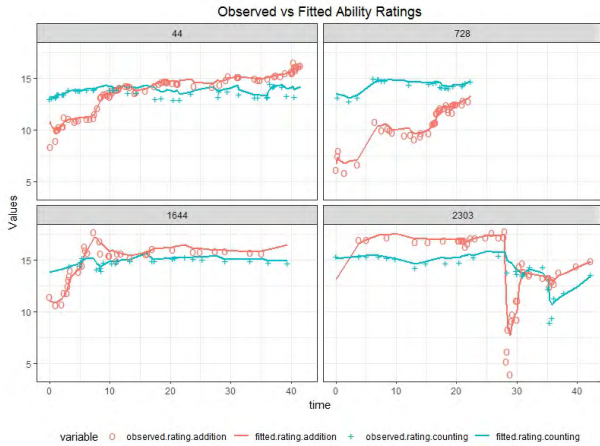


Figure 6: Observed and fitted skill ratings from the mutualism model.

actions between counting and addition ratings had a positive effect on their level changes. These results indicated that counting and addition skills collaborate, instead of competing, to form a positive manifold in the long run.

In summary, we have found beneficial interactions between children’s addition and counting skill ratings as being better at one skill helps being better at the other. The mutualism model was a better fit to the data than the g -factor model. Individual differences are present in the data in both starting positions of the change trajectories and key model parameters that represent limited resources in the system, providing potential evidence for both the g -theory and the mutualism model of general intelligence, according to [29]. We concur with van de Maas and colleagues (2006) that individual differences cannot be ignored in educational applications.

ferences cannot be ignored in educational applications.

5. CONCLUSIONS

In this paper, we presented a state-space expression of the continuous-time mutualism model proposed by [29] where individual differences, process noise, and measurement errors were taken into account. The mutualism model allowed us to tackle the underlying mechanism of the skill development from a micro perspective. We fitted the theoretical model to empirical data naturally collected online in authentic educational settings. Results showed that improvement in addition skill could positively influence the development in the counting domain, and vice versa. The better fit of the mutualism model to the data compared to the g -factor model suggested that the collaboration between the counting and addition skills in their co-development served as a better interpretation of the observed positive manifold.

The characteristics of the time series data in the current study are not uncommon in education as digital technology has transformed our way of collecting data about learning. The paper illustrates one way to fit dynamic models to the multivariate noisy irregularly spaced data that are rich in our real life. We appreciate the potential to apply the current method to different learning data to improve our understanding of cognitive and non-cognitive developments.

Nevertheless, this work has limitations that future work should aim to overcome. First, only two variables were considered in the current sample, while the mutualism model could be extended to multiple dimensions. The estimation algorithm is well suited for multivariate time series data, but the interpretation of the multivariate model can become complicated. Second, the estimation framework permits only a limited number of random effects in the current study [18]. In addition to the two carrying capacity parameters, one may be interested in adding random effects in the interaction parameters because of the potential competition between skills under time and attention constraints as we discussed above. The limitation of the estimation framework may be circumvented by utilizing sampling-based algorithms although they may be computationally heavy.

The fitting of the model to the data does not exclude other probable ways of interpreting cognitive development. Intervention studies with deliberate experimental designs are needed to establish causal relations in a dynamic system. These interventions may take the form of randomized assignment of skills to practice, for example, with groups of students assigned to practice only counting or only addition, but with progress measured on both skills after some period of practice. The cross-skill influence of practice can then be evaluated relative to practiced skill improvement.

Future work should also aim to evaluate how the mutualism account of skill development relates to other findings in education. For example, evidence suggests that interleaving practice on different problem types produces more robust learning and generalization than does blocking practice by problem type [31, 24]. It is possible that some of the benefit from interleaving relates to mutualism, with practice from different problem types influencing the development of the other skills.

6. ACKNOWLEDGMENTS

The work is supported by ACTNext by ACT, Inc.

7. REFERENCES

- [1] M. Begon, C. R. Townsend, and J. L. Harper. *Ecology: from individuals to ecosystems*. Number Sirsi) i9781405111171. 2006.
- [2] D. Borsboom, G. J. Mellenbergh, and J. V. Heerden. The Theoretical Status of Latent Variables. *Psychological Review*, 110(2):203–219, 2003.
- [3] H. Cen, K. Koedinger, and B. Junker. Comparing two irt models for conjunctive skills. In B. P. Woolf, E. Aïmeur, R. Nkambou, and S. Lajoie, editors, *Intelligent Tutoring Systems*, volume 5091, pages 796–798, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.
- [4] S.-M. Chow, L. Ou, A. Ciptadi, E. Prince, D. You, M. D. Hunter, J. M. Reh, A. Rozga, and D. S. Messinger. Representing sudden shifts in intensive dyadic interaction data using differential equation models with regime switching. *Psychometrika*, 83(2):476–510, 2018.
- [5] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [6] A. E. Elo. *The rating of chessplayers, past and present*. Arco Pub., 1978.
- [7] G. E. Gignac. Residual group-level factor associations: Possibly negative implications for the mutualism theory of general intelligence. *Intelligence*, 55:69–78, mar 2016.
- [8] F. E. Harrell Jr, with contributions from Charles Dupont, and many others. *Hmisc: Harrell Miscellaneous*, 2018. R package version 4.1-1.
- [9] A. D. Hofman, R. A. Kievit, C. E. Stevenson, D. Molenaar, I. Visser, and H. L. van der Maas. The dynamics of the development of mathematics skills: A comparison of theories of developing intelligence. *OSF Preprints*:<https://doi.org/10.31219/osf.io/xa2ft>, 2018.
- [10] I. Karatzas and S. E. Shreve. *Brownian Motion and Stochastic Calculus*. Springer New York, 1998.
- [11] R. A. Kievit, A. Hofman, and K. Nation. Mutualistic coupling between vocabulary and reasoning in young children: A replication and extension of kievit et al.(2017). *Psychological Science*, 2018.
- [12] S. Klinkenberg. High speed high stakes scoring rule: Assessing the performance of a new scoring rule for digital assessment. *Communications in Computer and Information Science*, 1(439):114–126, 2014.
- [13] S. Klinkenberg, M. Straatemeier, and H. L. Van der Maas. Computer adaptive practice of maths ability using a new item response model for on the fly ability and difficulty estimation. *Computers & Education*, 57(2):1813–1824, 2011.
- [14] K. Kovacs and A. R. A. Conway. Process Overlap Theory: A Unified Account of the General Factor of Intelligence. *Psychological Inquiry*, 27(3):151–177, 2016.
- [15] G. Y. Kulikov and M. V. Kulikova. Accurate numerical implementation of the continuous-discrete extended kalman filter. *IEEE Transactions on Automatic Control*, 59(1), January 2014.
- [16] A. J. Lotka. *Elements of Physical Biology*. Williams & Wilkins, Baltimore, MD, 1925.
- [17] G. Maris and H. Van der Maas. Speed-accuracy response models: Scoring rules based on response time and accuracy. *Psychometrika*, 77(4):615–633, 2012.
- [18] L. Ou. *Estimation of nonlinear mixed-effects continuous-time models*. PhD thesis, The Pennsylvania State University, 2018.
- [19] L. Ou, M. D. Hunter, and S.-M. Chow. *dynr: Dynamic Modeling in R*, 2017. R package version 0.1.11-5.
- [20] L. Ou, M. D. Hunter, and S.-M. Chow. What’s for dynr: A package for linear and nonlinear dynamic modeling in R. *The R Journal*, 2019.
- [21] Z. A. Pardos and N. T. Heffernan. Modeling individualization in a bayesian networks implementation of knowledge tracing. In P. De Bra, A. Kobsa, and D. Chin, editors, *User Modeling, Adaptation, and Personalization*, pages 255–266, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [22] P. I. Pavlik Jr., H. Cen, and K. R. Koedinger. Performance Factors Analysis — A New Alternative to Knowledge Tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education, (AIED09)*, pages 531–538, Amsterdam, 2009. IOS Press.
- [23] G. Rasch. *Probabilistic Models for Some Intelligence and Attainment Tests*. Studies in mathematical psychology. Danmarks Paedagogiske Institut, 1960.
- [24] D. Rohrer, R. F. Dedrick, and S. Stershic. Interleaved Practice Improves Mathematics Learning. *Journal of Educational Psychology*, 106(4):1–9, 2014.
- [25] C. Spearman. "General Intelligence," Objectively Determined and Measured Source : The American Journal of Psychology , Vol . 15 , No . 2 (Apr . , 1904), pp . 201-292 Published by : University of Illinois Press Stable URL : <https://www.jst. The American Journal of Psychology>, 15(2):201–292, 1904.
- [26] L. L. Thurstone. Primary mental abilities. *Psychometric monographs*, pages ix–121, 1938.
- [27] H. van der Maas, K.-J. Kan, and D. Borsboom. Intelligence is what the intelligence test measures. Seriously. *Journal of Intelligence*, 2(1):12–15, feb 2014.
- [28] H. Van Der Maas, K.-J. Kan, M. Marsman, and C. E. Stevenson. Network Models for Cognitive Development and Intelligence. *Journal of Intelligence*, 5(2):16, apr 2017.
- [29] H. L. van der Maas, C. V. Dolan, R. P. Grasman, J. M. Wicherts, H. M. Huizenga, and M. E. Raijmakers. A dynamical model of general intelligence: the positive manifold of intelligence by mutualism. *Psychological review*, 113(4):842, 2006.
- [30] V. Volterra. Fluctuations in the abundance of a species considered mathematically. *Nature*, 118:558–560, 1926.
- [31] Y. Weinstein, C. R. Madan, and M. A. Sumeracki. Teaching the science of learning. *Cognitive Research: Principles and Implications*, 3(2):1–17, 2018.
- [32] M. V. Yudelson. Individualizing bayesian knowledge tracing: are skill parameters more important than student parameters? *EDM*, 2016.

Detecting Wheel-spinning and Productive Persistence in Educational Games

V. Elizabeth Owen¹, Marie-Helene Roy¹, K. P. Thai¹, Vesper Burnett¹, Daniel Jacobs¹, Eric Keylor¹, Ryan S. Baker²

¹Age of Learning, 101 N. Brand Blvd, Glendale CA 91023

²Graduate School of Education, University of Pennsylvania, 3700 Walnut St., Philadelphia, PA 19104

v.elizabeth.owen@gmail.com, marie.roy@aofl.com, kpthai@gmail.com,
vesper.burnett@aofl.com, daniel.jacobs@aofl.com, eric.keylor@aofl.com,
ryanshaunbaker@gmail.com

ABSTRACT

Games in service of learning are uniquely positioned to offer immersive, interactive educational experiences. Well-designed games build challenge through a series of well-ordered problems or activities, in which perseverance is key for working through in-game failure and increasing game difficulty. Indeed, persistence through challenges during learning is beneficial not just in games but in other contexts as well, with grit and perseverance positively associated with academic performance and learning outcomes. However, recent studies suggest that not all persistence is positive, suggesting that many students end up “wheel-spinning”, spending considerable time on a topic without achieving mastery. Thus, it is vital to differentiate productive and unproductive persistence in order to understand emergent student progress, particularly in the context of learning games and personalized learning systems, in which individual pathways differ greatly based on student needs. Leveraging Educational Data Mining methods, this study builds a detector of wheel-spinning behavior (differentiated from productive persistence) in an adaptive, game-based learning system. With the ability to predict unproductive persistence early, this detection model can be used to intelligently adapt to students needing further support in-system, as well as informing in-person intervention in a classroom setting—thus supporting a personalized, engaging learning experience in both formal and informal learning environments.

Keywords

Behavior detection, predictive modeling, productive persistence, wheel-spinning, educational games, personalized learning

1. INTRODUCTION

Games as learning vehicles can offer engaging, interactive experiences in which the player has agency in exploring and solving well-ordered problems or challenges in a learner-responsive environment [1, 2]. Well-designed games seamlessly embed meaningful instruction in authentic, narrative-driven

learning contexts (with the potential to assess learning in the natural progression of play [3]). As such, they have the ability to optimize learner motivation and learning trajectories without removing the experience of personal discovery [4]. By nature, games encourage discovery of an underlying rule system through boundary testing, making experimentation and failure a core part of play progression [5]. In this sense, moving through in-game failure and challenge with perseverance can be fundamental to the experience of learning in games (e.g. [6, 7]). Hence, games offer a particularly relevant context for productive persistence or grit—the ability to steadily maintain an action or complete a task despite failure or adversity (cf. [8]). Indeed, keeping players in a “flow” state of persistence [9] through a series of challenges of increasing difficulty is key to the design of “good” games, particularly in educational contexts [10]. Recent research suggests these qualities in games support student growth in areas such as academic learning, socio-emotional skills, and creative problem solving (e.g. [11, 12, 13, 14]).

Indeed, persevering through challenges during learning is beneficial not just in games but in other contexts as well. From undergraduates to military cadets to Spelling Bee competitors, findings suggest that persistence forecasts strong performance in rigorous, achievement-based learning contexts [15]. In many cases, persistence is also associated with academic achievement [16], creativity [17], and long-term outcomes like earnings and later schooling [18].

However, recent research suggests that not all persistence is positive. “Wheel-spinning” is a form of unproductive effort, where students spend too much time struggling to learn a topic without achieving mastery [19]. Wheel-spinning behaviors have been associated with reduced motivation [20] and avoiding asking for help when needed [21]. In fact, recent empirical investigation has demonstrated that wheel-spinning can be differentiated from productive persistence in an intelligent tutoring system, in real-time, determining during problem-solving whether a student’s persistence will be productive [22]. Making this type of differentiation could also be valuable in learning games contexts. Persistence is important in games just as in other settings [23], with evidence suggesting that persisting unproductively in games can be a highly frustrating experience (e.g. [24]). Since challenge and problem solving are often core components of learning experiences, particularly in game-based environments, it becomes increasingly important to differentiate productive persistence (e.g. grit) from unproductive persistence (e.g. wheel-spinning) in the context of play. This differentiation could be used to offer different pathways to students based on real-time performance.

V. Elizabeth Owen, Marie-Helene Roy, K. P. Thai, Vesper Burnett, Daniel Jacobs, Eric Keylor and Ryan S. Baker “Detecting Wheel Spinning and Productive Persistence in Educational Games” In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 378 - 383

There is evidence that this type of modeling is feasible; related game-based research has shown that the same surface behavior in games can have different meanings, with distinction of productive vs unproductive failure in a games context (cf. [25]).

In this study, we empirically investigate wheel-spinning vs productive persistence in an adaptive, game-based learning system for early childhood math skills called *Mastering Math*. Specifically, we use predictive analytics to infer whether a student is engaging in wheel-spinning or productive persistence in *Mastering Math*. This detection model can be used to intelligently adapt to students needing further support in-system, as well as informing in-person intervention in a classroom setting—thus supporting a personalized learning experience in both formal and informal learning environments.

2. METHODS AND DATA COLLECTION

2.1 Game-based Learning Content

Mastering Math (MM) is a game-based adaptive learning system designed to help elementary age children build a strong understanding of fundamental number sense and operations, ranging from counting to 10 to adding and subtracting three-digit numbers using the standard algorithm. The app constitutes approximately 130 games, covering number sense and operations concepts and skills for pre-kindergarten through second grade. Each individual game maps to a learning objective, and is supported by an interactive instruction level, as well as several layers of scaffolding and feedback. In addition, the game system as a whole uses cohesive narrative and interactive characters (embedded at the level of individual games) to support student engagement with the learning world. Adaptivity functions within individual games to provide scaffolding with each level of skill difficulty, between games to adjust to students' difficulty needs, and across the system to give players a customized pathway between skills based on performance. Assessment is embedded throughout the play experience, including game-based pretests and final assessment tasks at a granular skill level.

2.2 Experimental Design

In the fall of 2018, two research studies were conducted to evaluate the effectiveness of *MM* in preschool (Study 1) and kindergarten (Study 2) students. Students in both studies came from ethnically diverse, low-income, public school districts in Southern California.

Both studies employed a cluster-randomized trial design, in which half of the participating classrooms in each study were randomly assigned to use the *MM* app as part of their classroom instruction (treatment group), while the other half used business-as-usual mathematics instruction and materials (control group). The treatment group students (394 students in total, 146 from Study 1, 248 from Study 2) were asked to use *MM* in small group settings for 15 minutes per day for three days per week, over a total of 12 weeks. After classroom implementation, overall usage averaged 5.6 hours in Study 1, and 5.22 hours in Study 2. Both treatment and control groups received a paper-and-pencil standardized assessment of early mathematics performance before and after the implementation of *MM*.

2.3 Event-stream Data Collection

Event stream data were collected using a learning game data framework based on ADAGE (Assessment Data Aggregator for

Game Environments; [26]), focusing on key learning mechanic milestones as context for performance information and results, as well as comprehensive coverage of player interaction and system feedback (e.g. [27]). These milestones are called *units*, and represent repeating progress mechanics through the learning game. Generally, students can play many games in the system (each of which corresponds with a mathematics skill), and each game contains multiple levels of difficulty. Thus, a larger unit of play is a game, and within a game a student can play one level (or activity) at a time. Each activity is built to support and assess knowledge of an individual math skill. All unit starts and ends are marked in the data, and all player interactions, system feedback, and results are recorded in the *context* of the active units at the time of the event. For example: if a student taps on the screen, we capture the basic x,y coordinate, the object being tapped (if applicable), and the units that were active during the tap (e.g. which game, activity and round the player was in when the event fired). In-game performance information (e.g. *result* or score) is embedded at the unit level, recorded at the end of applicable rounds and activities. In terms of raw *player interaction*, data collected consists of taps and drags. System feedback, also called *system events*, consists mainly of the game communicating with the player in giving formative feedback. This includes tutorial prompts, instructional input, and inactivity prompts (given if students have not interacted with the screen in 30 seconds). Additionally, every log file event is seeded with metadata such as student ID, timestamp, and session ID. Data structured in this fashion (Figure 1) allows for a comprehensive event-stream record that is labeled consistently across the system—which currently contains over 130 activities—all aligned with learning design for interpretability, a key element of viable data use for feature engineering and analysis.

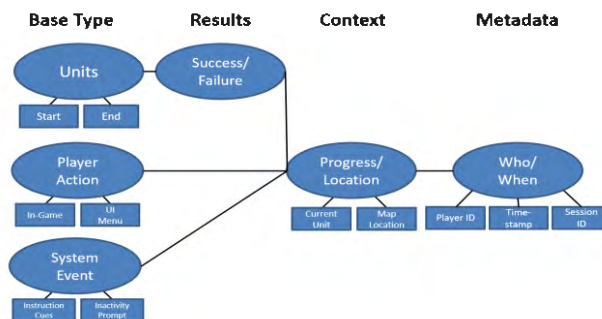


Figure 1. A simplified view of *MM*'s log file data schema.

2.4 Behavior Detection

To investigate player patterns of wheel-spinning in *Mastering Math*, prediction modeling was used to build a behavior detector (i.e. model of student behavior), an automated model that can infer from log files whether a student is behaving in a certain way (e.g. [28]). These models can be employed to detect a variety of important aspects of the learner and his/her performance, including student learning, strategy, and engagement (e.g. [29, 12, 30, 31]). To train the predictive model, detectors often leverage human judgment of student behavior, in a process where behavior labels derived from human judgement are used to train and validate models, which can then automatically detect the target behavior in the larger event-stream. In this case, once the initial student interaction with a digital learning environment is captured, the analysis process includes: 1) distilling data features potentially relevant to the behavior construct; 2) identifying

instances of the behavior through human evaluation; and 3) predictive modeling with the synchronized log file data.

Throughout these phases of analysis, a critical element of the data mining approach is emphasis on the event-stream trajectories that emerge in relationship to the behavior. With this detector study, each student's event-stream play patterns were observed and coded individually for emergent wheel-spinning behavior. Specific actions and click-stream interactions then emerged as evidence of wheel-spinning through the prediction algorithm's variable selection processes. Thus, player choices and interactions characteristic of wheel-spinning were derived from the larger event-stream data flow in the analysis process detailed below.

2.4.1 Feature Distillation

In this analysis, data features were distilled from *MM* event-stream data based on play across the entire system, then refined along themes of progression and performance. These organizing themes help capture student trajectories across the system for behavior detection, particularly since student progress and failure/success are central to the target constructs of wheel-spinning and productive persistence.

Using the learning progress mechanics, or units (i.e. games or activities) from the event-stream data schema, data features were organized based on performance within each unit, as well as measures of progression (e.g. time elapsed, number of activities completed, number of games activated, etc.). (For reference, when a game is activated, it means that a student failed the associated pretest and that gameplay for that skill is now open.) Summary features were also created in parallel to the unit features, giving a sense of the overall trajectory of the player through the learning space. Since PreK and K students are in developing stages of cognition, additional features were engineered to represent age and elements of motor skill (e.g. miss rate, or how often a student drags an object towards a target and misses). One view of selected event-stream features is given below (Table 1).

Table 1. Overview of selected event-stream features

	<i>Progression</i>	<i>Performance</i>
<i>Overall</i>	<ul style="list-style-type: none"> total duration in system (active play) total activities completed total activities started total games started miss rate student age 	<ul style="list-style-type: none"> % of skills mastered ratio of "boss" activities successfully passed* total # of skills (games) activated total skills mastered (games completed)
<i>Game</i>	<ul style="list-style-type: none"> game completion rate avg duration to game completion # of answers submitted per player per game # of activities completed within each game avg time elapsed between activities in the same game 	<ul style="list-style-type: none"> individual game status: <ul style="list-style-type: none"> - in-progress - passed game (skill completed) - struggling (fail states for 3 of the last 5 activities) - not started - pretest passed** - pretest failed % of started games successfully completed
<i>Activity</i>	<ul style="list-style-type: none"> activity completion rate avg activity duration # of hints given # of inactivity prompts # of tutorials accessed 	<ul style="list-style-type: none"> score progression to next level (pass/fail) # of rounds passed # of rounds failed # of rounds completed

*The "boss" activity is the most difficult assessment in a game

**Pretests are embedded at the game level to test prior knowledge

2.4.2 Behavior Coding of Wheel-spinning

For behavior detection, we focused on the construct of wheel-spinning, since the ability to flag this particular behavior held strong utility for enabling automated scaffolding in-system as well as in-person teacher intervention. Wheel-spinning is also an especially relevant focus for a game context—a medium in which boundary testing is an implicit norm [25], and differentiating real struggle from more productive forms of exploration and self-paced discovery can be valuable. *Mastering Math* games are sufficiently different from the intelligent tutoring systems, where wheel-spinning was initially studied, to require a different operationalization of wheel-spinning. In this context, we view wheel-spinning as connected to lower gameplay efficiency in the system, since wheel-spinning occurs when a great deal of effort yields very little progress [19]. To capture efficiency in an adaptive games context, in which every student has a different learning pathway, we designed a metric allowing efficiency to be standardized across players. This measure of learning efficiency was called *rate of mastery*, designed to measure the rate at which students were mastering math skills. This was calculated as the number of boss activities (the hardest assessment level in each math skill game) a student passed, divided by his/her total number of activities. This measure made sense as a progress-based metric, since performance on boss-level skill assessments is central to learning game progression. This ratio was ultimately calculated using data from both school studies. In the main behavior analysis, in accordance with the focus on wheel-spinning students and those persevering through difficulty, we concentrated on students in the lower two quartiles of *rate of mastery*.

As noted in Kai et al., 2018, we cannot assume that all lower efficiency students in the system are hopelessly struggling—on the contrary. Students who take their time to learn material, use self-paced progression, and achieve eventual success are likely to be demonstrating productive persistence. Determining whether a low-efficiency student is spinning their wheels or persisting productively is challenging. To differentiate these two groups, we started by leveraging human judgement on a per-student level to capture emergent patterns in the data. In particular, we chose to utilize the human capacity for pattern recognition and behavior evaluation (rather than an a priori rule-based approach), since the system is adaptive and no two students are likely to have the same path through the learning space.

Thus, the next step was to have human researchers observe a stream of student actions and identify the student's behavior (e.g. [32]). The human evaluation of student behavior establishes when the behavior occurred (which serves as the predicted variable). For coding of wheel-spinning behavior in this study, play visualization based on text replays were adopted for their efficiency and accuracy [33]. Text replays, based on recorded log file data, are a text-based representation of student action during a given period of time. Text replays have shown to be highly time-efficient and scalable [50], and almost as accurate for detecting student behavior as other methods such as live observation [34].

The variation on the text replay that we used—called a visual progress replay (VPR)—includes color coding of performance levels (in addition to text summaries) for greater ease of information processing (cf. [35]). This approach represents the same information as a text replay, but in a form that encodes information with color consistent with canonical visualization

techniques [36], and has previously been used to create detection models in related game-based learning research (e.g. [25]). Key features of the VPR included a visual display of game status, per student, across the system. This color-coded visual map showed whether each game was in-progress, completed (passed), in a struggle state (i.e. at least 3 non-passing scores within the last 5 activities), or not started. In addition, summary statistics per student were shown, such as number of activities completed in the system and time spent in total (Figure 2).

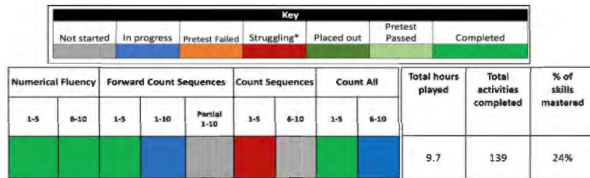


Figure 2. A sample portion of a VPR used for coding wheel-spinning, shown for a single student across the system.

We designed the replay's clip size to show one student's full system playthrough at a time, since we wanted to be able to detect a system-wide wheel-spinning state for each child. To capture the full trajectory of play, coding was done at the student level, labeling each student at the end of the study (week 12), in terms of whether a student was WS (wheel-spinning), P (productive persistence), and NA (not enough information). Within the lower efficiency group of students, WS captured a state of high effort but little progress, P fit with steady student progress, and NA was applied when there wasn't enough information (e.g. not enough time or activities in game to make a judgement). We included NA in this schema so that we could *derive* time and activity minimums for WS vs P differentiation through the predictor itself, rather than picking an arbitrary cutoff in excluding student data (such as, for instance, dropping all students who played for under 30 minutes). This third code also allowed for more nuanced coding—rather than forcing all students to fit under WS or P, thus risking miscategorization, the NA code could be used instead. Using this tri-code schema, inter-rater reliability analysis yielded a Cohen's κ [37] of .78, indicating acceptable agreement between raters was achieved.

2.4.3 Modeling Early Detection of Wheel-spinning

The final predictive model merged the initial feature engineering of event-stream features with the behavioral codes generated in the analysis above. To support early intervention for in-system personalization as well as teacher interventions, the final model was built to predict wheel-spinning (WS) at the end of week 12 (the last week of the study) using predictors from week 4 data. (Week 4 predictors were selected after subsequent weeks 5 and 6 were tested for model performance, but resulted in only marginal improvement.) Since each classroom was assigned exactly 12 weeks of play relative to start date, weeks as a time marker helped consistently align student progress across classrooms in relationship to the study design. It also allowed for implementation-focused behavior detection for the highest utility to teachers. With earlier detection of students getting stuck in the system, intervention can have greater impact on student progress in building core math skills.

Ultimately, the log file features (Table 1) were used as predictors in the model, while the behavior of wheel-spinning became the predicted variable. Using this full feature list, the WS detector was then built at the student level using RStudio, using the RWeka

package for data mining [38]. An appropriate set of algorithms were selected based on the categorical dependent variable, informed by related behavior modeling research in education (e.g. [39, 40]), including J48, CART, Random Forest, and Naïve Bayes. Models were evaluated using ten-fold cross validation, with a final selection based on the goodness metric of AUC ROC.

To achieve higher accuracy in correctly detecting and classifying the target class of students, the wheel-spinning students, we used rebalanced classes for all three methods tested. This approach is in alignment with similar detector-based analyses in digital learning contexts (e.g. [41, 42]). Specifically, the original classes P, WS and NA had a respective number of instances of 79, 39 and 14. We set the target number of instances for each rebalanced class to $n=100$. To obtain that number of observations per class, for a total of $n=300$, sampling with replacement was performed on each class. This resampling procedure was only performed on the training set for tree building purposes and all testing was performed on the original data distribution.

3. RESULTS AND DISCUSSION

3.1 Results

Ultimately, the CART algorithm produced the best model performance, achieving a cross-validated AUC of .676 in predicting wheel-spinning in week 12 with week 4 predictors, comparable to metrics in other game-based learning detector models (e.g. [39, 12]).

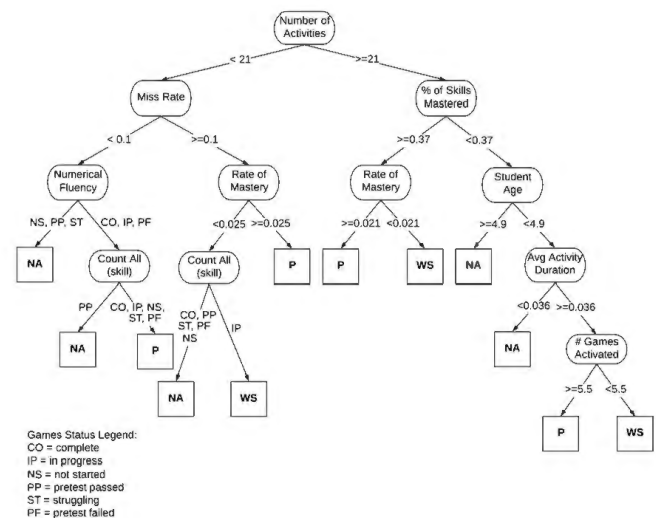


Figure 3. Final CART wheel-spinning predictor model.

Interestingly, in Figure 3 the first decision on the tree is the number of activities completed, with many students having less than 21 activities categorized as NA (insufficient information). Under this parent node, three core pathways emerge for wheel-spinners, covered from left to right on the tree: 1) low prior knowledge and low motor skill (via count all game, a low-level skill, and miss rate); 2) higher % of skills mastered but with very low efficiency (rate of mastery); and 3) younger students with low prior knowledge and very low efficiency in the system (with few games activated, higher time per activity and high number of activities per game). The first path suggests a group of students that may be in an earlier stage of development, both in terms of motor skill and prior knowledge (but not necessarily age). The second group, with the ability to master more skills with seemingly fewer motor skill issues, may represent students that

have more advanced development and prior knowledge but may need a bit more scaffolding and just-in-time support to learn the material. The last group may consist of younger students with low prior knowledge who need support more directly related to age-based maturity levels. These groups, implying differing levels of development and age, reflect clinical research which suggests wide variation in the relationship between age and developmental level in young children [43]. This suggests that developmental stage (rather than age alone) is a helpful differentiator in personalizing learning experiences for young students. To investigate implications of age and development for better learning design, these emergent groups suggest value in deeper exploration of student profiles in future work (discussed below). Overall, these model-derived groups offer insight into potential types of wheel-spinning that occur within the system, with the tree model allowing for early detection of unproductive persistence.

3.2 Discussion and Conclusion

Overall, the model of wheel-spinning yields insight into the important differentiation between unproductive and productive persistence, revealing multiple ways that student wheel-spinning manifests in data and enabling event-stream detection of this behavior in the event stream data. In turn, this real-time prediction can allow for very early intervention—both in-system and in classroom—for students displaying wheel-spinning behavior. For the system, this means it may be possible to offer more intelligent adaption to student needs, while for teachers (with limited time and resources) it may become possible to offer just-in-time information about which students most need help. This emergent behavior detection is especially important in games, which can have unexpected player pathways due to complex elements of narrative, agency, and failure-driven exploration—all of which converge to support the medium’s power of engagement in well-designed playful learning experiences.

Along this line of research in future work, there is an opportunity to generalize this detector to children using the game-based learning system outside of a study-specific context. The week-level data used in this study was centered around implementation, designed to flag to a teacher which students might be wheel-spinning after a certain amount of prescribed weekly dosage; however, converting this progress marker to an activity/elapsed time-based unit to build a model based on data in the wild can make this model applicable to an even broader base of learners. In addition, comparison between the classroom-based and broader event-stream based models may yield interesting insights. There is also an opportunity in this rich data stream (currently thousands of students) to hone the model for even higher AUC and predictive power. This includes iteration in feature engineering based on patterns that may arise in the larger data stream of students, using predictive modeling of wheel-spinning in a broader context of students (in formal and informal learning environments). Investigating player profiles based on detection results may also help determine groups of students struggling with the system based on motor skill, prior knowledge, and age/grade. Relatedly, better understanding how motor skill indicators in the data connected to more traditional measures of visual-motor skill (e.g. [44]) may also be valuable. Finally, dashboards highlighting detector-based insights to both parents and students for interpersonal support represent a key area for future work, with potential for student-level flagging for intervention, specific skills needing support (see Figure 2), recommendations for in-person

follow-up, and possible grouping of students in the same class for differentiated instruction.

Future work in expanding the scope of wheel-spinning research in the *MM* system can support the ability to generalize findings across broader age ranges and geographic areas, increasing the potential for impact on data-driven design, intelligent personalization, and interpersonal intervention. With information on behaviors like wheel-spinning and productive persistence, in combination with other evidence such as student prior knowledge, this work can inform designers about which instructional design in games needs revisiting, as well as providing adaptive logic and system overlays for just-in-time detection and intervention. Both in the system and beyond, this research can further the application of educational data mining to principled learning design, potentially expanding the field of intelligent game-based learning and supporting young learners in developing foundational academic skills at scale.

4. ACKNOWLEDGMENTS

We would like to thank the Mastery team at Age of Learning, including Doug Dohring, Sunil Gunderia, and Diana Hughes.

5. REFERENCES

- [1] J. P. Gee, “Learning by design: Good video games as learning machines,” *E-Learn. Digit. Media*, vol. 2, no. 1, pp. 5–16, 2005.
- [2] K. Squire, “From content to context: Videogames as designed experience,” *Educ. Res.*, vol. 35, no. 8, pp. 19–29, 2006.
- [3] V. J. Shute, “Stealth assessment in computer-based games to support learning,” in *Computer Games and Instruction*, S. Tobias and J. D. Fletcher, Eds. Charlotte, NC: IAP, 2011, pp. 503–524.
- [4] L. P. Rieber, “Seriously considering play: Designing interactive learning environments based on the blending of microworlds, simulations, and games,” *Educ. Technol. Res. Dev.*, vol. 44, no. 2, pp. 43–58, 1996.
- [5] K. Salen and E. Zimmerman, *Rules of play: Game design fundamentals*. Cambridge, MA: MIT Press, 2004.
- [6] K. Bielaczyc and M. Kapur, “Playing epistemic games in science and mathematics classrooms,” 2010.
- [7] V. J. Shute *et al.*, “Modeling how incoming knowledge, persistence, affective states, and in-game progress influence student learning from an educational game,” *Comput. Educ.*, vol. 86, pp. 224–235, Aug. 2015.
- [8] A. L. Duckworth and P. D. Quinn, “Development and Validation of the Short Grit Scale (Grit-S),” *J. Pers. Assess.*, vol. 91, no. 2, pp. 166–174, Feb. 2009.
- [9] M. Csikszentmihalyi, *Flow: The psychology of optical experience*. New York: Harper Perennial, 1990.
- [10] J. P. Gee, “Big ‘G’ Games,” <http://www.jamespaulgee.com/node/63>, 2012. .
- [11] J. Hamari, D. J. Shernoff, E. Rowe, B. Coller, J. Asbell-Clarke, and T. Edwards, “Challenging games help students learn: An empirical study on engagement, flow and immersion in game-based learning,” *Comput. Hum. Behav.*, vol. 54, pp. 170–179, Jan. 2016.
- [12] R. S. Baker and J. Clarke-Midura, “Predicting Successful Inquiry Learning in a Virtual Performance Assessment for Science,” in *Proceedings of the 21st International Conference on User Modeling, Adaptation, and Personalization*, New York, NY, 2013, pp. 203–214.

- [13] T. R. A. Kral *et al.*, “Neural correlates of video game empathy training in adolescents: a randomized trial,” *Npj Sci. Learn.*, vol. 3, no. 1, p. 13, Aug. 2018.
- [14] C. Steinkuehler and S. Duncan, “Scientific Habits of Mind in Virtual Worlds,” *J. Sci. Educ. Technol.*, vol. 17, no. 6, pp. 530–543, Sep. 2008.
- [15] A. L. Duckworth, C. Peterson, M. D. Matthews, and D. R. Kelly, “Grit: Perseverance and passion for long-term goals,” *J. Pers. Soc. Psychol.*, vol. 92, no. 6, pp. 1087–1101, 2007.
- [16] A. E. Poropat, “A meta-analysis of the five-factor model of personality and academic performance,” *Psychol. Bull.*, vol. 135, no. 2, pp. 322–338, 2009.
- [17] V. Prabhu, C. Sutton, and W. Sauser, *Creativity and Certain Personality Traits: Understanding the Mediating Effect of Intrinsic Motivation*, vol. 20. 2008.
- [18] J. Deke and J. Haimson, “Valuing Student Competencies: Which Ones Predict Postsecondary Educational Attainment and Earnings, and for Whom?,” p. 150.
- [19] J. E. Beck and Y. Gong, “Wheel-Spinning: Students Who Fail to Master a Skill,” in *Artificial Intelligence in Education*, vol. 7926. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 431–440.
- [20] G. Sedek and M. Kofta, “When cognitive exertion does not yield cognitive gain: Toward an informational explanation of learned helplessness,” *J. Pers. Soc. Psychol.*, vol. 58, no. 4, pp. 729–743, 1990.
- [21] J. T. Dillon, *Questioning and Teaching: A Manual of Practice*. New York: Teachers College, 1988.
- [22] S. Kai, M. V. Almeda, R. S. Baker, C. Heffernan, and N. Heffernan, “Decision Tree Modeling of Wheel-Spinning and Productive Persistence in Skill Builders,” *J. Educ. Data Min.*, vol. 10, no. 1, pp. 36–71, 2018.
- [23] K. E. DiCerbo, “Game-Based Assessment of Persistence,” *Educ. Technol. Soc.*, vol. 17, no. 1, pp. 17–28, 2014.
- [24] V. E. Owen, D. Ramirez, A. Salmon, and R. Halverson, “Capturing Learner Trajectories in Educational Games through ADAGE (Assessment Data Aggregator for Game Environments): A Click-Stream Data Framework for Assessment of Learning in Play,” presented at AERA, Apr-2014.
- [25] V. E. Owen, G. Anton, and R. S. Baker, “Modeling User Exploration and Boundary Testing in Digital Learning Games,” in *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, New York, NY, 2016, pp. 301–302.
- [26] R. Halverson and V. E. Owen, “Game Based Assessment: An Integrated Model for Capturing Evidence of Learning in Play,” *Int. J. Learn. Technol. Spec. Issue Game-Based Learn.*, vol. 9, no. 2, pp. 111–138, 2014.
- [27] V. E. Owen and R. S. Baker, “Fueling Prediction of Player Decisions: Foundations of Feature Engineering for Optimized Behavior Modeling in Serious Games,” *Technol. Knowl. Learn.*, vol. 24, pp. 1–26, 2018.
- [28] R. S. Baker, A. T. Corbett, and K. R. Koedinger, “Detecting student misuse of intelligent tutoring systems,” in *Intelligent tutoring systems*, 2004, pp. 531–540.
- [29] M. A. Sao Pedro, R. S. Baker, J. D. Gobert, O. Montalvo, and A. Nakama, “Leveraging machine-learned detectors of systematic inquiry behavior to estimate and predict transfer of inquiry skill,” in *User Modeling and User-Adapted Interaction*, 2013, pp. 1–39.
- [30] K. E. DiCerbo and K. Kidwai, “Detecting Player Goals from Game Log Files,” in *Proceedings of the 6th International Conference on Educational Data Mining*, Massachusetts, USA, 2013, pp. 314–316.
- [31] J. Asbell-Clarke, E. Rowe, and E. Sylvan, “Assessment design for emergent game-based learning,” in *CHI’13 Extended Abstracts on Human Factors in Computing Systems*, New York, NY, 2013, pp. 679–684.
- [32] R. S. Baker and A. de Carvalho, “Labeling student behavior faster and more precisely with text replays,” in *Proceedings of the 1st International Conference on Educational Data Mining*, Massachusetts, USA, 2008, pp. 38–47.
- [33] R. S. Baker, “Mining data for student models,” in *Advances in intelligent tutoring systems*, Springer, 2010, pp. 323–337.
- [34] R. S. Baker, A. T. Corbett, and A. Z. Wagner, “Human classification of low-fidelity replays of student actions,” in *Intelligent Tutoring Systems*, New York, NY, 2006, pp. 29–36.
- [35] J. A. Wise *et al.*, “Visualizing the non-visual: Spatial analysis and interaction with information from text documents,” in *Information Visualization*, 1995, pp. 51–58.
- [36] E. R. Tufte and P. R. Graves-Morris, *The visual display of quantitative information*, vol. 2. Cheshire, CT: Graphics press, 1983.
- [37] J. Cohen, “A coefficient of agreement for nominal scales,” *Educ. Psychol. Meas.*, vol. 20, no. 1, pp. 37–46, 1960.
- [38] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The WEKA data mining software: an update,” *ACM SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.
- [39] S. Kai, M. V. Almeda, R. S. Baker, N. Schectman, C. Heffernan, and N. Heffernan, “Modeling Wheel-spinning and Productive Persistence in Skill Builders,” in *Proceedings of the 10th International Conference on Educational Data Mining*, Massachusetts, USA, 2017.
- [40] M. Wixon, R. S. Baker, J. D. Gobert, J. Ocumpaugh, and M. Bachmann, “WTF? detecting students who are conducting inquiry without thinking fastidiously,” in *User Modeling, Adaptation, and Personalization*, Springer, 2012, pp. 286–296.
- [41] N. Bosch, Y. Chen, and S. D’Mello, “It’s Written on Your Face: Detecting Affective States from Facial Expressions while Learning Computer Programming,” in *Intelligent Tutoring Systems*, vol. 8474. Cham: Springer International Publishing, 2014, pp. 39–44.
- [42] S. Kai *et al.*, “A Comparison of Video-Based and Interaction-Based Affect Detectors in Physics Playground,” in *Proceedings of the 8th International Conference on Educational Data Mining*, Massachusetts, USA, 2015, pp. 77–84.
- [43] J. Squires, D. Bricker, and L. Potter, “Revision of a Parent-Completed Developmental Screening Tool: Ages and Stages Questionnaires,” *J. Pediatr. Psychol.*, vol. 22, no. 3, pp. 313–328, 1997.
- [44] K. E. Beery, “The Beery-Buktenica developmental test of visual-motor integration.” NCS Pearson., Minneapolis, MN, 2004.

A Self-Attentive model for Knowledge Tracing

Shalini Pandey
University of Minnesota
Twin Cities, MN 55455, USA
pande103@umn.edu

George Karypis
University of Minnesota
Twin Cities, MN 55455, USA
karypis@umn.edu

ABSTRACT

Knowledge tracing is the task of modeling each student's mastery of knowledge concepts (KCs) as (s)he engages with a sequence of learning activities. Each student's knowledge is modeled by estimating the performance of the student on the learning activities. It is an important research area for providing a personalized learning platform to students. In recent years, methods based on Recurrent Neural Networks (RNN) such as Deep Knowledge Tracing (DKT) and Dynamic Key-Value Memory Network (DKVMN) outperformed all the traditional methods because of their ability to capture a complex representation of human learning. However, these methods face the issue of not generalizing well while dealing with sparse data which is the case with real-world data as students interact with few KCs. In order to address this issue, we develop an approach that identifies the KCs from the student's past activities that are *relevant* to the given KC and predicts his/her mastery based on the relatively few KCs that it picked. Since predictions are made based on relatively few past activities, it handles the data sparsity problem better than the methods based on RNN. For identifying the relevance between the KCs, we propose a self-attention based approach, Self Attentive Knowledge Tracing (SAKT). Extensive experimentation on a variety of real-world dataset shows that our model outperforms the state-of-the-art models for knowledge tracing, improving AUC by 4.43% on average.

Keywords

Knowledge Tracing, Massive Open Online Courses, Self-attention, sequential recommendation

1. INTRODUCTION

The availability of massive dataset of students' learning trajectories about their *knowledge concepts* (KCs), where a KC can be an exercise, a skill or a concept, has attracted data miners to develop tools for predicting students' performance and giving proper feedback [8]. For developing such person-

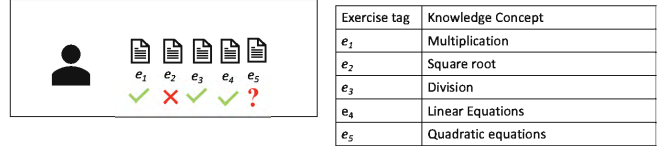


Figure 1: Left subfigure shows the sequence of exercises that the student attempts and the right subfigure shows the knowledge concepts to which each of the exercises belong.

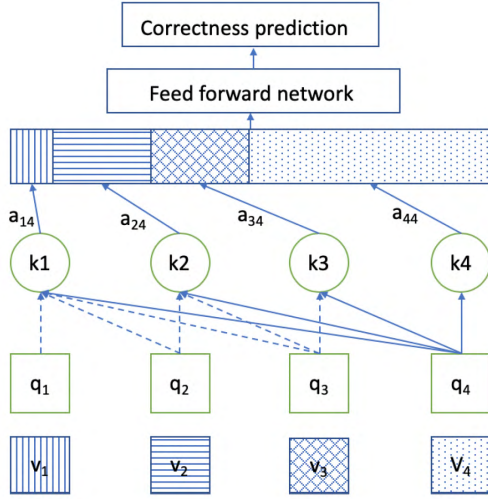
alized learning platforms, knowledge tracing (KT) is considered to be an important task and is defined as the task of tracing a student's *knowledge state*, which represents his/her mastery level of KCs, based on his/her past learning activities. The KT task can be formalized as a supervised sequence learning task - given student's past exercise interactions $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t)$, predict some aspect of his/her next interaction \mathbf{x}_{t+1} . On the question-answering platform, the interactions are represented as $\mathbf{x}_t = (e_t, r_t)$, where e_t is the exercise that the student attempts at timestamp t and r_t is the correctness of the student's answer. KT aims to predict whether the student will be able to answer the next exercise correctly, i.e., predict $p(r_{t+1} = 1 | e_{t+1}, \mathbf{X})$.

Recently deep learning models such as Deep Knowledge Tracing (DKT) [6] and its variant [10] used Recurrent Neural Network (RNN) to model a student's knowledge state in one summarized hidden vector. Dynamic Key-value memory network (DKVMN) [11] exploited Memory Augmented Neural Network [7] for KT. Using two matrices, *key* and *value*, it learns the correlation between the exercises and the underlying KC and student's knowledge state, respectively. The DKT model faces the issue of its parameters being non-interpretable [4]. DKVMN is more interpretable than DKT as it explicitly maintains a KC representation matrix (*key*) and a knowledge state representation matrix (*value*). However, since all these deep learning models are based on RNNs, they face the issue of not generalizing while dealing with sparse data [3].

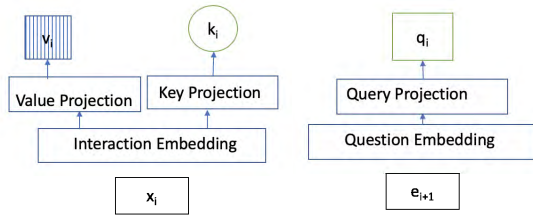
In this paper, we propose to use a purely attention mechanism based method, *transformer* [9]. In the KT task, the skills that a student builds while going through the sequence of learning activities, are related to each other and the performance on a particular exercise is dependent on his performance on the past exercises related to that exercise. For example, in figure 1, for a student to solve an exercise on "Quadratic equation" (exercise 5) which belongs to the knowl-

Shalini Pandey and George Karypis "A Self Attentive model for Knowledge Tracing" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 384 - 389

edge concept “Equations”, he needs to know how to find “square roots” (exercise 3) and “linear equations” (exercise 4). SAKT, proposed in this paper first identifies *relevant* KCs from the past interactions and then predicts student’s performance based on his/her performance on those KCs. For predicting student’s performance on an exercise, we used exercises as KCs. As we show later, SAKT assigns weights to the previously answered exercises, while predicting the performance of the student on a particular exercise. The proposed SAKT method significantly outperforms the state-of-the-art KT methods gaining a performance improvement of 4.43% on the AUC, on an average across all datasets. Furthermore, the main component (self-attention) of SAKT is suitable for parallelism; thus, making our model order of magnitude faster than RNN based models.



(a) Network of SAKT. At each timestamp the attention weights are estimated for each of the previous element only. Keys, Values and Queries are extracted from the embedding layer shown below. When j th element is query and i th element is key, attention weight is $a_{i,j}$.



(b) Embedding layer embeds the current exercise that the student is attempting and his past interactions. At every time stamp $t+1$, the current question e_{t+1} is embedded in the query space using Exercise embedding and elements of past interactions x_t is embedded in the key and value space using the Interaction embedding.

Figure 2: Diagram showing the architecture of SAKT.

2. PROPOSED METHOD

Our model predicts whether a student will be able to answer the next exercise e_{t+1} based on his previous interaction sequence $\mathbf{X} = x_1, x_2, \dots, x_t$. As shown in figure 2, we can transform the problem into a sequential modeling

Table 1: Notations

Notations	Description
N	total number of students
E	total number of exercises
\mathbf{X}	Interaction sequence of a student: (x_1, x_2, \dots, x_t)
x_i	i th exercise-answer pair of a student
n	maximum length of sequence
d	latent vector dimensionality
\mathbf{e}	Sequence of exercises solved by the student
\mathbf{M}	Interaction embedding matrix
\mathbf{P}	Positional embedding matrix
\mathbf{E}	Exercise lookup matrix
$\hat{\mathbf{M}}$	Past interactions embedding
$\hat{\mathbf{E}}$	Exercise embedding

problem. It is convenient to consider the model with inputs x_1, x_2, \dots, x_{t-1} and the exercise sequence with one position ahead, e_2, e_3, \dots, e_t and the output being the correctness of the response to exercises r_2, r_3, \dots, r_t . The interaction tuple $x_t = (e_t, r_t)$ is presented to the model as a number $y_t = e_t + r_t \times E$, where E is the total number of exercises. Thus, the total values that an element in the interaction sequence can take is $2E$, while elements in the exercise sequence can take E possible values.

We now describe the different layers of our architecture.

Embedding layer: We transform the obtained input sequence $\mathbf{y} = (y_1, y_2, \dots, y_t)$ into $s = (s_1, s_2, \dots, s_n)$, where n is the maximum length that the model can handle. Since the model can work with inputs of fixed length sequence, if the sequence length, t is less than n , we repetitively add a *padding* of question-answer pair to the left of the sequence. However, if t is greater than n , we partition the sequence into subsequences of length n . Specifically, when t is greater than n , y_t is partitioned into t/n subsequences each of length n . All these subsequences serve as input to the model.

We train an *Interaction embedding matrix*, $\mathbf{M} \in \mathbb{R}^{2E \times d}$, where d is the latent dimension. This matrix is used to obtain an embedding, \mathbf{M}_{s_i} for each element, s_i in the sequence. Similarly, we train exercise embedding matrix, $\mathbf{E} \in \mathbb{R}^{E \times d}$ such that each exercise in the set e_i is embedded in the e_i th row.

Position Encoding: Position Encoding is the layer in the self-attention neural network which is used for encoding the position so that like convolution network and recurrent neural network, we can encode the order of the sequence. This layer is particularly important in knowledge tracing problem because a student’s knowledge state evolves gradually and steadily with time. The knowledge state at a particular time instance should not show wavy transitions [10]. In order to incorporate this we use a parameter, position embedding, $\mathbf{P} \in \mathbb{R}^{n \times d}$ which is learned while training. The i th row of position embedding matrix, \mathbf{P}_i is then added to the interaction embedding vector of the i th element of the interaction sequence.

The output from the embedding layer is embedded interac-

tion input matrix, $\hat{\mathbf{M}}$ and embedded exercise matrix, $\hat{\mathbf{E}}$:

$$\hat{\mathbf{M}} = \begin{bmatrix} \mathbf{M}_{s_1} + \mathbf{P}_1 \\ \mathbf{M}_{s_2} + \mathbf{P}_2 \\ \dots \\ \mathbf{M}_{s_n} + \mathbf{P}_n \end{bmatrix}, \quad \hat{\mathbf{E}} = \begin{bmatrix} \mathbf{E}_{s_1} \\ \mathbf{E}_{s_2} \\ \dots \\ \mathbf{E}_{s_n} \end{bmatrix}. \quad (1)$$

Self-attention layer: In our model, we use the scaled dot-product attention mechanism [9]. This layer finds the relative weight corresponding to each of the previously solved exercise for predicting the correctness of the current exercise.

We obtain query and key-value pairs using the following equations:

$$\mathbf{Q} = \hat{\mathbf{E}}\mathbf{W}^Q, \mathbf{K} = \hat{\mathbf{M}}\mathbf{W}^K, \mathbf{V} = \hat{\mathbf{M}}\mathbf{W}^V, \quad (2)$$

where $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{d \times d}$ are the query, key and value projection matrices, respectively, which linearly project the respective vectors to different space [9]. The relevance of each of the previous interactions with the current exercise is determined using the attention weights. For finding the attention weights we use the scaled dot product [9], defined as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}. \quad (3)$$

Multiple heads: In order to jointly attend to information from different representative subspaces, we linearly project the queries, keys and values h times using different projection matrices.

$$\text{Multihead}(\hat{\mathbf{M}}, \hat{\mathbf{E}}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)\mathbf{W}^O, \quad (4)$$

where $\text{head}_i = \text{Attention}(\hat{\mathbf{E}}\mathbf{W}_i^Q, \hat{\mathbf{M}}\mathbf{W}_i^K, \hat{\mathbf{M}}\mathbf{W}_i^V)$ and $\mathbf{W}^O \in \mathbb{R}^{hd \times d}$.

Causality:

In our model, we should consider only first t interactions when predicting the result of the $(t+1)$ st exercise. Therefore, for a query \mathbf{Q}_i , the keys \mathbf{K}_j such that $j > i$ should not be considered. We use, causality layer to mask the weights learned from a future interaction key,

Feed Forward layer:

The self-attention layer described above results in weighted sum of values, \mathbf{V}_i of the previous interactions. However the rows of the matrix obtained from the multihead layer, $\mathbf{S} = \text{Multihead}(\hat{\mathbf{M}}, \hat{\mathbf{E}})$ is still a linear combination of the values, \mathbf{V}_i of the previous interactions. To incorporate non-linearity in the model and consider the interactions between different latent dimensions, we use a feed forward network.

$$\mathbf{F} = \text{FFN}(\mathbf{S}) = \text{ReLU}(\mathbf{S}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})\mathbf{W}^{(2)} + \mathbf{b}^{(2)}, \quad (5)$$

where $\mathbf{W}^{(1)} \in \mathbb{R}^{d \times d}$, $\mathbf{W}^{(2)} \in \mathbb{R}^{d \times d}$, $\mathbf{b}^{(1)} \in \mathbb{R}^d$, $\mathbf{b}^{(2)} \in \mathbb{R}^d$ are parameters learned during training.

Residual Connections: The residual connection [2] are used to propagate the lower layer features to the higher layers. Hence, if low layer features are important for prediction, the residual connection will help in propagating them to the final layers where the predictions are performed. In the context of KT, students attempt exercises belonging to

a specific concept to strengthen that concept. Hence, residual connection can help propagating the embeddings of the recently solved exercises to the final layer making it easier for model to leverage the low layer information. A residual connection is applied after both self-attention and feed forward layer.

Layer normalization: In [1], it was shown that normalizing inputs across features can help in stabilizing and accelerating neural networks. We used layer normalization in our architecture for the same purpose. Layer normalization is also applied at both the self-attention and feed forward layer.

Prediction layer:

Finally, each row of the matrix \mathbf{F}_i obtained above is passed through the fully connected network with Sigmoid activation to predict the performance of the student.

$$p_i = \text{Sigmoid}(\mathbf{F}_i \mathbf{w} + \mathbf{b}), \quad (6)$$

where p_i is a scalar and represents the probability of student providing correct response to exercise e_i , \mathbf{F}_i is the i th row of \mathbf{F} and $\text{Sigmoid}(z) = 1/(1 + e^{-z})$

Network Training: The objective of training is to minimize the negative log likelihood of the observed sequence of student responses under the model. The parameters are learned by minimizing the cross entropy loss between p_t and r_t .

$$\mathcal{L} = -\sum_t (r_t \log(p_t) + (1 - r_t) \log(1 - p_t)) \quad (7)$$

3. EXPERIMENTAL SETTINGS

3.1 Datasets

To evaluate our model, we used four real-world datasets and one synthetic dataset.

- *Synthetic*¹: This dataset is obtained by simulating 4000 virtual students' answering trajectories. Each student answers the same sequence of 50 exercises, which are drawn from 5 virtual concepts with varying difficulty level.
- *ASSISTment 2009*² (*ASSIST2009*): This dataset is provided by ASSISTment online tutoring platform and is widely used for KT tasks. We conducted our experiments on the updated "skill-builder" dataset. The dataset is sparse as the density of this dataset is 0.06, shown in Table 2.
- *ASSISTment 2015*³ (*ASSIST2015*): ASSISTment 2015 contains students' responses on 100 skills. There are 19,917 students and 708,631 interactions. Although the number of records in this dataset is more than ASSISTment 2009, the average number of records per student is smaller because the number of students is larger. This dataset is the most sparse of all the available datasets, with a density of 0.05.

¹<https://github.com/chrispiech/DeepKnowledgeTracing/tree/master/data/synthetic>

²<https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data/skill-builder-data-2009-2010>

³<https://sites.google.com/site/assistmentsdata/home/2015-assistments-skill-builder-data>

Table 2: Dataset Statistics

Datasets	#Users	#Skill tags	#Interactions	#Unique Interactions	Density
Synthetic-5	4000	50	200K	200K	1
ASSIST2009	4417	124	328K	35K	0.06
ASSIST2015	19917	100	709K	102K	0.05
ASSIST-Chall	686	102	943K	57K	0.81
STATICS	333	1223	190K	129K	0.31

The columns corresponding to #Users, #Skill tags and #Interactions represent the number of students, total number of exercise tags and the number of records, respectively. The column Density represents the density of each dataset (i.e., $\text{Density} = \frac{\text{\#Unique Interactions}}{(\text{\#Users} \times \text{\#Skill tags})}$).

- *ASSISTment Challenge (ASSISTChall)*: This data is obtained from ASSISTment 2017 competition⁴. It is the richest dataset in terms of the number of interactions with 942,816 interactions, 686 students and 102 skills. This dataset is the most dense dataset of all the available datasets because its density is 0.81.
- *STATICS2011 (STATICS)*: This dataset contains the interaction from an engineering statics course with 189,927 interactions, 333 students and 1223 skill tags. We adopted the processed data from [11]. It is also a dense dataset with a density of 0.31.

The complete statistical information for all the datasets can be found in Table 2.

3.2 Evaluation Methodology

Metrics: The prediction task is considered in a binary classification setting i.e., answering an exercise correctly or not. Hence, we compare the performance using the Area Under Curve (AUC) metric.

Approaches: We compare our model against the state-of-the-art KT methods, DKT [6], DKT+ [10], and DKVMN [11]. These methods are described in the introduction.

Model Training and parameter selection: We trained the model with 80% of the dataset and test it on the remaining. For all the methods, we tried the hidden state dimension $d = \{50, 100, 150, 200\}$. For the competing approaches, we used the same hyperparameters as reported in their respective papers. For initialization of weights and optimization, we used a similar procedure as [10]. We implemented SAKT with *Tensorflow* and used ADAM [5] optimizer with learning rate of 0.001. We used a batch size of 256 for the ASSISTChall dataset and 128 for the others. For datasets with a larger number of records, e.g., ASSISTChall and ASSIST2015, we used a dropout rate of 0.2, while for the remaining datasets, we used a dropout rate of 0.2. We set the maximum length of the sequence, n as roughly proportional to the average exercise tags per student. For ASSISTChall and STATICS dataset we use $n = 500$, for the ASSIST2009 $n = 100$ and 50, for the synthetic and ASSIST2015 datasets n is set to 50.

⁴<https://sites.google.com/view/assistmentsdatamining>

Table 3: Student Performance prediction comparison.

Datasets	AUC				
	DKT	DKT+	DKVMN	SAKT	Gain%
Synthetic	0.823	0.824	0.822	0.832	0.97
ASSIST2009	0.820	0.822	0.816	0.848	3.16
ASSIST2015	0.736	0.737	0.727	0.854	15.87
ASSISTChall	0.734	0.728	0.689	0.734	0.00
STATICS	0.815	0.835	0.814	0.853	2.16
Average	0.786	0.789	0.773	0.824	4.43

¹ **Bold** numbers are the best performance.

² The reported results are obtained by the best hyperparameter selection for each dataset individually.

4. RESULTS AND DISCUSSION

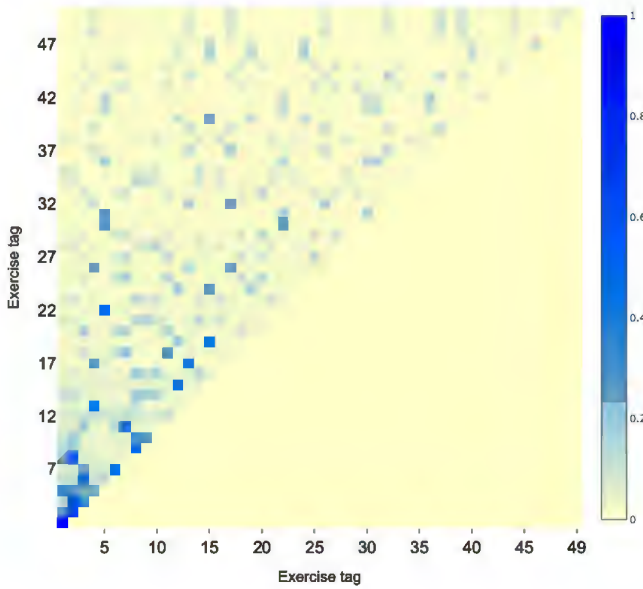
Student Performance Prediction: Table 3 shows the performance comparison of SAKT with the current state-of-the-art methods. On the Synthetic dataset, SAKT performs better than the competing approaches, achieving an AUC of 0.832 compared to 0.824 by DKT+. Even though Synthetic is the most dense dataset, SAKT outperforms RNN based methods because of the methodology used for generating Synthetic. For this dataset, each individual exercise is derived from only one concept. The probability of a student answering an exercise from this dataset correctly is determined using Item Response Theory [8] as, $p(\text{correct}|\alpha, \beta) = c + \frac{1-c}{1+\exp(\beta-\alpha)}$, where c denotes the probability of guessing it correctly, α and β are randomly chosen numbers to indicate the concept ability and exercise difficulty, respectively. Thus, in this dataset, the exercises belonging to the same concept are strongly correlated. SAKT, unlike other benchmarks, directly attempts to identify exercises belonging to the same concept and hence performs better than other methods. On ASSIST2009, SAKT performs better than competing approaches, gaining a performance improvement of 3.16% over the second best performing method. For ASSIST2015 dataset, SAKT shows an impressive improvement of 15.87%. We attribute this gain to the fact that attention mechanism leveraged by SAKT can learn and generalize well even when the dataset is sparse, which is the case with ASSIST2015 as its density is the least among the other datasets. For STATICS2011, our method achieves a performance improvement of 2.16% compared to DKT+. For ASSISTChall, our method performs at par with DKT. This can be attributed to the fact that ASSISTChall is the most dense dataset of all the real-world datasets.

Attention weights visualization: Visualizing the attention weights between the elements of past interactions (which serve as keys) and the exercise that the student is going to solve next (which serves as query) can help in understanding which exercises in the past interactions are relevant to the query exercise. With this motivation, we compute the sum of attention weights of each exercise pair ($e1, e2$) across all the sequences where $e1$ serves as query and interaction with exercise $e2$ serves as key. We then normalize the attention weights so that the sum of the weights for each query is one. This results in a *relevance* matrix in which each element, ($e1, e2$) represents the influence of $e2$ on $e1$. We perform our analysis on Synthetic because this dataset was generated with known hidden concepts and hence the ground

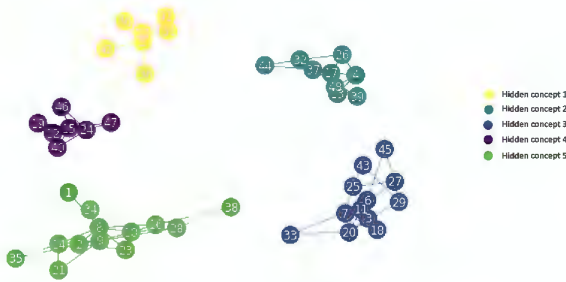
Table 4: Example of attention weights for some sequences in ASSIST2009 dataset.

Exercise tag	Past Interactions
Scale Factor	(Probability of Two Distinct Events,1): 0.000001, (Circle Graph, 1): 0.0001, (Circle Graph,1):0.001, (Division Fractions, 0): 0.99
Ordering integers	(Intercepts,0): 0.21, (Ordering positive decimals,1): 0.611 , (Multiplication whole numbers,1): 0.09, (Proportion,1):0.033
Rate	(Interior Angles Figures, 0):0.005, (Algebraic Simplification,0) : 0.009, (Rate,0):0.5 , (Interior Angles Figures, 0):0.1, (Algebraic Simplification,0) : 0.12

The columns corresponding to Exercise tag refers to the query (i.e., the exercise for which we have to predict the student's performance) and Past Interactions refers to the sequence of interactions that has been observed for that student, respectively. **red** colored elements in the right column represent the most important element among the past interaction elements



(a) Heatmap depicting the attention weights between each pair of exercises. Note that, the weight assigned for pair (i, j) , where $j > i$ is always zero because all the sequences consists of exercises in the same order from



(b) Graph depicting the relevance between exercises. The relevance is determined by the attention weights learned between the exercises using SAKT. We observe a perfect clustering of latent concepts.

Figure 3: Visualizing attention weight of Synthetic dataset.

truth regarding the relevance of different exercises are known to us. Figure 3a shows the heatmap corresponding to the *relevance* matrix of exercises in Synthetic. For Synthetic, all the sequences consist of all exercise tags in the same sequence starting from 1 to 50.

In order to build the influence graph between the exercise tags, as shown in Figure 3b, we use the *relevance* matrix. Firstly, we draw out the first exercise in the sequence that belongs to each hidden concept, and visit each row of the *relevance* matrix, and connect the exercise corresponding to that row to the first two exercises ranked based on edge weight, which is proportional to the attention weights between the pair of exercises. We can see that the based on the attention weights, we are able to achieve the perfect clustering of the exercise tags based on the hidden concepts from which they are derived. An interesting observation is that two exercises which occur far apart in the sequence but belonging to the same concept can be identified by SAKT. For example, as shown Figure 3b a query on exercise 22 assigned most weight to the key with exercise 5 even when they occur far apart in the sequence.

Two exercises which are relevant to each other tend to have high attention weights as the performance on one of them impacts the performance on the other. Additionally, in the real-world scenario, the exercises which occur close in the sequence tend to belong to the same concept. Thus, we expect that the attention weights biased towards the exercises that occur recently in the interaction sequence. To illustrate this, we manually analyzed ASSIST2009 dataset to visualize the attention weights for some selected samples. Table 4 shows some of the exercises along with the past interactions and attention weights assigned to each interaction.

Ablation Study: Table 4 shows the performance of default SAKT architecture and all the variants on all the datasets (with $d = 200$).

No Positional Encoding (PE): In this variant of the default architecture, we removed the positional encoding. As a result, the attention weights assigned for predicting the performance of student on a particular exercise depends only on the interaction embedding, without being affected by its position in the sequence. In case of ASSIST2009 and ASSIST2015, the dataset is sparse and hence the impact of removal of PE is not much pronounced as is the case with the dense dataset such as ASSISTChall and STATICS.

No Residual Connection (RC): RCs shows the importance of low level features i.e., the interaction embedding while making the prediction. Since our architecture is not very deep, the RC do not contribute much to the performance of the model. In fact removal of residual connection gives better

Table 5: Ablation Study

Architecture	Synthetic	ASSIST 2009	ASSIST 2015	ASSIST Chall	STATICS
Default	0.832	0.848	0.854	0.734	0.853
No PE	0.827	0.842	0.849	0.715	0.832
No RC	0.823	0.847	0.857	0.709	0.834
No Dropout	0.832	0.845	0.851	0.711	0.840
Single head	0.823	0.828	0.845	0.709	0.851
0 block	0.826	0.837	0.822	0.634	0.819
2 blocks	0.827	0.840	0.853	0.724	0.845

performance than default for the ASSIST2015 dataset.

No Dropout: Dropout is used in neural network to regularize the model so that it can generalize better. Overfitting of the model is more effective for dataset with less number of records compared to the number of parameters of model. As a result, role of dropout is more effective for ASSIST2009 dataset and STATICS dataset.

Single head: Instead of using 5 heads as is the case in default architecture, we tried a variant of using only one head. Multiple heads help in capturing the attention weights in different subspaces. Using single head consistently drops the performance of SAKT on all the datasets.

No block: When no self-attention block is used the prediction of the next exercise depends only on the last interaction. It can be seen that without attention block the performance is significantly worse than that of default architecture.

2 Blocks: Increasing the number of blocks of self-attention increases the number of parameters of the model. However, in our case this increase of parameters does not prove to be useful in improving the performance. The reason being an important aspect of prediction of performance of student at an exercise is dependent on his performance on the past relevant exercises. Adding another block of self-attention makes the model more complex.

Training efficiency: Figure 4 demonstrates the efficiency of various methods based on their run times on GPU during the training phase. Comparing the computational efficiency, SAKT only spends 1.4 seconds in one epoch which is 46.42 less than the time taken by DKT+ (65 seconds/epoch), 32 times less than DKT (45 seconds/epoch) and 17.33 times less than DKVMN (26 seconds/epoch). We conducted the experiments on a single GPU of type NVIDIA Titan V.

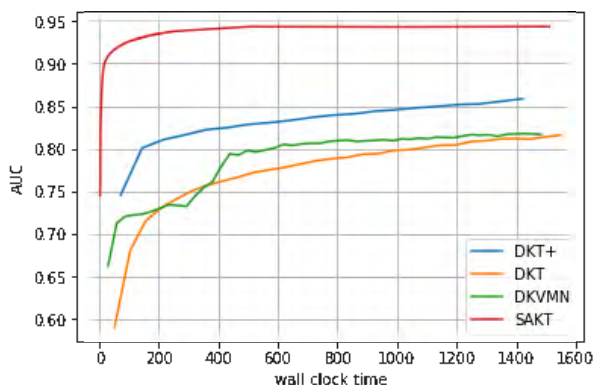


Figure 4: Training Efficiency on ASSIST2009 dataset.

5. CONCLUSION AND FUTURE WORK

In this work, we proposed a self-attention based knowledge tracing model, SAKT. It models a student's interaction history (without using any RNN) and predicts his performance on the next exercise by considering the relevant exercises from his past interactions. Extensive experimentation on a variety of real-world datasets shows that our model can outperform the state-of-the-art methods and is an order of magnitude faster than the RNN-based approaches.

6. ACKNOWLEDGEMENT

This work was supported in part by NSF (1447788, 1704074, 1757916, 1834251), Army Research Office (W911NF1810344), Intel Corp, and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute.

7. REFERENCES

- [1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).
- [2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
- [3] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. *CoRR* abs/1808.09781 (2018).
- [4] Mohammad Khajaj, Robert V Lindsey, and Michael C Mozer. 2016. How deep is knowledge tracing? *arXiv preprint arXiv:1604.02416* (2016).
- [5] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
- [6] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*. 505–513.
- [7] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065* (2016).
- [8] John Self. 1990. Theoretical foundations for intelligent tutoring systems. *Journal of Artificial Intelligence in Education* 1, 4 (1990), 3–14.
- [9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.
- [10] Chun-Kit Yeung and Dit-Yan Yeung. 2018. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. *arXiv preprint arXiv:1806.02180* (2018).
- [11] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*.

Success prediction in MOOCs

A case study

Antoine Pigeau, Olivier Aubert and Yannick Prié
LS2N, University of Nantes, France
firstName.lastName@univ-nantes.fr

ABSTRACT

Success prediction in Massive Open Online Courses (MOOCs) is now tackled in numerous works, but still needs new case studies to compare the solutions proposed. We study here a specific dataset from a French MOOC provided by the OpenClassrooms company, featuring 12 courses. We exploit various features present in the literature and test several classification models.

1. INTRODUCTION

Multiple models and data mining methods for learner success prediction in a Massive Open Online Courses (MOOCs) are proposed by many works in the literature [1], with different conclusions about which model provides the best performance. The quality of the results seems to highly depend on the input dataset, and on the selected or computed features. Generalization of the methodology for success prediction seems now ongoing [1], but we still need new case studies to improve the accuracy and insights obtained by these methods.

This work presents a case study on a new dataset, provided by OpenClassrooms, a major online courses french company. We test several models using classification algorithms and sequence-based approaches, such as process and pattern mining. Our study aims at enriching previous results obtained for different datasets published in the literature.

Our first contribution in this work is the comparison of 8 classification models. Random Forest, AdaBoost, Support Vector Machine (SVM), logistic regression and neural networks are first applied, followed by sequence-based approaches: an LSTM neural network, a process mining method and a proposal of a solution based on a sequence mining method. The second contribution consists in experimental results obtained from a new dataset for a success prediction task. While most papers only focus on 1-5 courses [1], we use here 12 different courses from the same platform.

Antoine Pigeau, Olivier Aubert and Yannick Prié "Success prediction in MOOCs - A case study" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 390 - 395

The remainder of this paper is organized as follows: section 2 surveys previous work related to success prediction. Section 3 presents the dataset used for the experiments with details on the raw data used to compute features. Section 4 describes features obtained from the literature and their adaptation to our context. Section 5 presents the classification methods that we applied. The result of our experiments are detailed and discussed in section 6. We conclude by summarizing our work and drawing perspectives in section 7.

2. RELATED WORK

Prediction of dropout or success in MOOCs is carried out in numerous works [1]. The goal is to improve the performance of the learners by detecting a possible failure in advance. Such a detection could for instance lead to a teacher intervention to increase the learner engagement in the course.

The input data for such a prediction is based either on assignments [2, 3, 4, 5, 6] or clickstreams [7, 8, 9, 10]. Social activities can also be included to assess the learner engagement in a course [2, 3, 5].

Classification methods rely on common approaches such as linear regression, logistic regression, K-nearest neighbors, random forests, decision trees, support vector machines, hidden Markov models and neural networks. Because the course context is different for each study, it is hard to determine which model will be the best for a prediction task. Support vector machine is the best method obtained for [5] while random forest performs better in [6]. The conclusion of [2] states that prediction performance depends more on the features computed than on the model.

Several temporal data mining methods are proposed in the literature [1, 8, 9]. Recurrent neural networks are assessed for dropout detection and experiments conclude that LSTM recurrent networks present the second best results in [11] (where a Nonlinear State Space Model is slightly better). [9] proposes a solution based on process mining to emphasize a correlation between the way learners browse the course and their performances. Other approaches use sequence mining algorithms to predict learner skills [8].

In the present work, we test some of the common shallow methods proposed in the literature, as well as neural network approaches. We also explore a solution based on process mining, and propose one based on sequence mining. For all

Name	# users	# pass	# fail
Java	7761	34 (0%)	7727 (100%)
XML	855	10 (1%)	845 (99%)
Ionic	960	46 (1%)	914 (99%)
Rubys	149	5 (3%)	144 (97%)
Node JS	2227	81 (4%)	2146 (96%)
Arduino	2487	115 (5%)	2372 (95%)
Bootstrap	8402	727 (8%)	7675 (92%)
Audace Entr.	225	26 (12%)	199 (88%)
JavaScript	8105	1803 (22%)	6302 (78%)
Gestion Projet	1808	666 (36%)	1142 (64%)
Twitter	817	328 (40%)	489 (60%)
Web	7947	3502 (44%)	4445 (56%)

Table 1: Number of learners per group for each course of the OpenClassrooms dataset, after cleanup. The first column presents the total number of learners and the two last ones detail the number of passing and failing learners.

these algorithms, we assess a large set of features adapted from the literature.

3. INPUT DATA

OpenClassrooms is a MOOC platform that provides courses in various domains, from art and culture to computer science. All courses are freely accessible anytime, and paid services are proposed for supplementary features such as online help and certificates of achievement. Courses are generally composed of texts, videos or e-books that users can browse, read or download after a registration process. Based on the properties proposed in [1], these courses can be characterized as follows:

- massive, open and online: thousands of learners can follow the courses freely. Paid access is provided to get an access to a tutor or a completion certificate;
- no-stakes: the learner can complete a course without certification or credit;
- asynchronous: learners are free to register, browse the content, or complete a course. There is no constraint on dates for enrollment or assignments. This point is important and has an impact on the choice of features used as input for the prediction model;
- heterogeneous: learners have various motivations and mostly come from francophone countries since courses are in french.

In this study, we are considering a dataset covering 12 courses in the domains of programming languages, project management and startup creation. The two leftmost columns of table 1 present for each course its name and the total number of learners that followed it, from 2014 to 2016.

The provided courses are composed of static web pages and quizzes/assignments, and do not contain any video. A course is composed of chapters, divided in sections, and of exercises based on quizzes and assignments. To succeed in a course, a learner must obtain an average grade on all exercises higher than 70/100 (the exercises are quizzes automatically graded).

The input format of learner activity is a clickstream dataset. Each access to a resource is recorded as one event in a log file. The granularity of the retrieved events varies among different courses. Apart from Audace Entreprendre, Node JS, XML and Java (in bold in table 1), which are traced at the section level, all courses present a chapter granularity.

The first step applied on the raw dataset consists in segmenting the learners' sequences of events into sessions. This session detection step aims to enrich some features related to the learners' regularity, the duration, or the number of events in the working sessions. The learners' sessions are determined from the raw sequences with the method proposed in [12], where a session is defined as *a delimited and sustained set of pages visited by the same user within the duration of one particular visit to a particular website*. Once sessions are determined, a cleaning task is performed: learners with only one session and no exercise attempt are removed. We associate this behaviour to learners that want to check the content of the course and do not really intend to follow it.

Discussion with the OpenClassrooms company about their needs lead us to define two groups for our goal of success prediction:

- passing group: set of learners that obtained an average grade equal to or higher than 70/100 for a course;
- failing group: set of learners that did not obtain an average grade higher than 70/100. This group contains all the learners that either quit the course or completed all exercises but failed to obtain a grade higher than the 70/100 validation threshold.

Our choice for the terms "Pass / Fail" is based on [1], where it is defined that *A student typically passes a course if they meet or exceed an instructor-specified overall grade threshold; otherwise they fail*.

The two rightmost columns of table 1 present the number of learners in each group for the 12 courses of the dataset, after the cleaning step. As commonly encountered in MOOC contexts, these groups are clearly unbalanced: on average, the passing group represents 15% of the learners. Courses of table 1 are ordered according to the percentage of passing learners.

4. FEATURES

Table 2 presents our candidate features set. This set regroups an adaptation of the best features identified in [2, 7, 13, 14, 15]. We needed to adapt some of the features due to differing contexts for our MOOC. A set of regularity features proposed in [10] was also used in our experiments: the features PDH, PWD, WS1, WS2, WS3, FDH, FWH and FWD were tested for our classification task. Check [10] for more details on these features.

Several options are possible to generate features depending on the considered machine learning approach:

- basic features: features are computed for the whole considered period (after x weeks of the course for instance). These features do not evolve with time;
- temporal features: features are computed for succes-

Category	name	description	abbrev.
Activity	total duration	total time spent on the course	totalDuration
	window duration	time spent from the first event to the last one	windowDuration
	linearity	ratio between the number of correct transitions, w.r.t. the structure of the course and the number of transitions in the sequence	linearityRatio
	# events per resource	# of accesses per resource	#eventPerR
	duration per resource	total time spent on each resource	durationPerR
	average duration per resource	avg duration spent on each resource	avgDurationPerR
	average session duration	avg duration of user sessions	avgSessionDuration
	average # events in session	avg number of events in each user session	avg#eventSession
Inter-activity periods	median of intertime	median of the time spent between sessions	medianInterTime
	duration before assessment*	time spent before each assessment	durationBeforeA
	# events before assessment*	# of events before each assessment	#eventBeforeA
	# sessions before assessment*	# of sessions between each assessment	#sessionBeforeA
	# events per day before assessment*	average number of events per day between each assessment	#eventPerDayBeforeA
	# sessions per day before assessment*	avg # of sessions per day between each assessment	#sessionPerDayBeforeA
	time since last event*	time without activity after 1, 2,..., n weeks (n=7 weeks in our experiment)	timeSinceLastEvent
Assignment	marks*	all marks obtained for each quiz	marks

Table 2: Features used for our experiments: this set is composed of features identified in our literature review, adapted to the characteristics of our dataset (* indicates the features defined by a set of values).

sive time periods in order to emphasize their evolution all along the course. The period commonly used in the literature seems to be one week [1];

- temporal features with stacking: similar to the previous method but each feature of a period is stacked with the previous one. Practically, it consists in adding the values of week n with those of week $n+1$.

5. PREDICTION TASK

In this section, we present different classification methods tested for our prediction task of passing/failing. We start with the baseline methods commonly applied for this kind of task and then detail a process mining approach and our proposal based on a sequence mining solution.

5.1 Baseline approaches

In order to compare our results with other available works in the literature, we experimented with the following methods: Random Forest, AdaBoost, SVM, logistic regression, dense neural network and LSTM neural network.

A first step of feature selection is necessary for logistic regression and SVM models. We rely here on a wrapper method with a forward selection to emphasize the best features. A subset of features is iteratively built, starting from an empty set and adding one by one the features that best improve our model's accuracy for the whole set of courses. The process is stopped when accuracy does not increase anymore.

Except for LSTM neural network that directly relies on a sequence of features, other methods can deal with several

types of input features: basic features, temporal features and temporal features with stacking. We test each possibility in our experiment, to determine in what measure this choice impacts the performance of the prediction.

5.2 Process mining approach

Process mining was initially a method to analyze business processes for process discovery, process conformance checking and process improvement. In the context of online courses, this method proposes to study the behavior of learners during a course, by emphasizing common paths in course resource navigation.

The classifier for our prediction task is built from the outputs of process discovery and conformance checking methods. Our process discovery relies on the Heuristic Miner algorithm [16]. This algorithm is robust, and deals with the majority of common problems in process detection. Models for failing and passing are built with this algorithm for each course. Our conformance checking solution relies on an algorithm based on an alignment method [16]. Our prediction task is carried out by computing the fitness of a learner on both failing and passing models, and affecting him to the group with the best fitness.

The input dataset of a process mining algorithm is a set of traces, where each trace represents the sequence of activities of one learner. In our context, an activity is an access to a resource and is defined with the id of this resource. Because the grades and the duration of each access are lost, a categorization step is carried out on each event of a learner's trace. It consists in updating the resource ids (the activity) as follows: for an exercise id, the new id depends on success or failure. For a chapter/section, the new id depends

Sequence length	Random forest	AdaBoost	SVM	Logistic regression
25%	marks, durationPerR, #sessionPerDayBeforeA, #eventBeforeA, avgDurationPerR	marks, avg#eventSession, durationPerR, avgDurationPerR	marks, #eventPerR, #sessionPerDayBeforeA, durationBeforeA	#eventPerR, marks, durationBeforeA, #sessionBeforeA, #eventPerDayBeforeA
50%	marks, #sessionPerDayBeforeA, #eventPerDayBeforeA	marks, avgDurationPerR, durationPerR, #eventPerR	#eventPerR, marks, #eventBeforeA, #eventPerDayBeforeA	#sessionBeforeA, #eventPerR, marks, #eventBeforeA
75%	marks, #sessionPerDayBeforeA, #eventPerDayBeforeA	marks, avgDurationPerR, durationPerR	marks, #eventPerDayBeforeA, #eventPerR, #sessionBeforeA	marks, #eventPerR
100%	marks	marks	marks	marks

Table 3: Best features for the different sequence lengths for random forest, AdaBoost, SVM and logistic regression models. This result is an aggregation of the best features obtained on each course separately. For each model and sequence length, features are ordered from most to least pertinent.

on the duration spent on the resource, using 3 classes of short/medium/long durations.

5.3 Sequence mining approach

With this approach, our goal is to determine whether different groups of learners present distinct frequent sub-sequences of events in their traces.

Our first step is to build the passing model (by retrieving the frequent sequence on only the passing learners) and the failing model. We rely on the VMSP algorithm to generate the maximal frequent sequences on both groups. Note that all sub-sequences of a maximal sequence are also frequent sequences, thus we still obtain all the frequent sequences.

Our second step is to compute a similarity score between a model and a learner's sequence. Our proposal consists in tessellating the new learner sequence with the larger frequent sub-sequences of the models. Practically, we try to map each frequent sequence on the learner sequence. The mapping obtained is used to compute a similarity score:

1. for a frequent sequence of length n in the model, generate all k -grams with k between 2 and n ;
2. map all k -grams one by one on the learner sequence, keeping the mapping with the larger k -gram;
3. repeat the steps 1-2 for all frequent sequences;
4. for each item position of the learner sequence, a score is computed as the length of the longer k -gram that maps this position. The similarity score is then obtained by summing up all these positions' scores.

The input dataset for this method is similar to the one used for our process mining approach. Each learner's event is categorized with our previous method (see section 5.2).

6. EXPERIMENTS

For all the following experiments, a cross validation 80% train - 20% test is carried out 10 times on each course separately. For the neural network approaches, the training set is divided into a train set, a validation set and a test set. Input features are standardized. The computation of temporal features is carried out by grouping the sessions into 7 days

periods. For each period, all features, except the regularity ones, are computed.

Neural networks present the advantage of avoiding the laborious feature selection step, but still need some tuning for determining a correct architecture with its optimization parameters. Our first task was to assess several candidate architectures, varying the number of layers and units. Our prediction tasks were carried out on all courses and results were aggregated. The best accuracies were obtained with the following parameters: [Dense Layer of 512 units, Dropout layer, Dense Layer of 256 units, Dropout layer, Dense layer of 1 unit with a sigmoid activation].

A similar search was carried out for the architecture of the LSTM solution, leading to the following parameters: [LSTM layer of 32 units, Dense layer of 1 units with a sigmoid activation]. The input of the LSTM algorithm, a time series, was computed as follows: each session is considered as a time step. For a specific learner, the input features for time t is computed with the learner t^{th} session and each element of a learner's time step is stacked with its previous element (a padding is applied to provide the same time's series length for each learner).

Finally, in order to assess our prediction at different time steps of the learning process, classification tasks are tested on truncated versions of the sequences. Experiments provide results for 25%, 50%, 75% and 100% of learner's sequence length (number of events).

In the following, the best features for random forest, logistic regression, SVM and AdaBoost models are first presented. Second, the results of the prediction task are detailed for each model.

6.1 Best features selection

The best features obtained for the shallow methods are presented in table 3. The best features for SVM and logistic regression are obtained with the wrapper method described in section 5.1. Best features on each course were computed with a 10 times 80%-20% cross validation, leading to a score for each feature depending on its ranking. These scores were

		25%			50%			75%			100%		
		A.	P.	F.	A.	P.	F.	A.	P.	F.	A.	P.	F.
Basic features	Random forest	91%	28%	96%	93%	36%	96%	94%	47%	96%	98%	77%	98%
	AdaBoost	91%	41%	94%	93%	48%	95%	94%	59%	96%	99%	96%	99%
	SVM	88%	61%	90%	91%	64%	92%	92%	69%	93%	98%	89%	99%
	Logistic Reg.	88%	66%	89%	91%	71%	91%	93%	84%	93%	99%	100%	99%
	Dense NN	91%	37%	94%	93%	47%	95%	94%	59%	95%	98%	82%	99%
	LSTM	90%	38%	93%	91%	47%	93%	93%	55%	95%	97%	78%	98%
	Process mining	66%	47%	64%	72%	49%	71%	67%	54%	65%	64%	54%	58%
	Seq. mining	24%	85%	13%	24%	88%	9%	22%	95%	5%	18%	100%	0%
Temporal features	Random forest	92%	9%	99%	92%	11%	99%	92%	12%	98%	92%	11%	99%
	AdaBoost	90%	42%	94%	92%	46%	95%	92%	46%	96%	95%	55%	98%
	SVM	88%	60%	90%	89%	65%	90%	92%	72%	92%	99%	87%	99%
	Logistic Reg.	87%	58%	91%	91%	67%	92%	92%	76%	92%	99%	90%	99%
	Dense NN	89%	36%	93%	90%	41%	93%	92%	48%	94%	94%	59%	96%
Temporal features with stacking	Random forest	89%	16%	97%	90%	22%	98%	91%	24%	97%	92%	29%	96%
	AdaBoost	91%	43%	94%	92%	47%	95%	93%	51%	96%	97%	67%	99%
	SVM	91%	66%	93%	87%	23%	95%	91%	57%	92%	91%	61%	92%
	Logistic Reg.	89%	64%	90%	90%	61%	92%	92%	76%	92%	98%	91%	98%
	Dense NN	88%	36%	92%	83%	18%	88%	79%	17%	82%	78%	18%	83%

Table 4: Accuracies of the different models tested. A., P. and F. stand respectively for All, Pass and Fail

then aggregated among the courses. For random forest and AdaBoost, the weights provided by the learning algorithms have been used. For each method and sequence length, we selected the features with the best scores until a sudden drop appeared (the elbow method).

Clearly, the number of best features decreases with the increase of the sequence length available for the classification task, leading to the sole use of marks (feature marks) for full length. The marks feature is obviously pertinent for all sequence lengths.

If we ignore marks, random forest, logistic regression and SVM seem more related to inter-activity periods features while AdaBoost is associated to activity features. The best features concern mainly the marks, the activity intensity and the activity intensity between assessments. No regularity feature appears in the best features list for any model.

In the following, the aggregated best features obtained for SVM and logistic regression models are used to provide the results of our prediction task.

6.2 Best models

Table 4 presents the aggregated accuracies obtained for each model on each course separately, with basic features, temporal features and temporal features with stacking.

Each row is associated to a model and the columns present the sequence length used to fit the model (25%, 50%, 75% and 100%). The sub-columns (A., P., F.) stand for All, Pass and Fail, respectively for the overall accuracy, the accuracy for the passing learners and the accuracy for the failing learners.

Among the shallow classification methods, the Adaboost and logistic regression models present the more balanced results on both the passing and failing groups: Adaboost

seems more reliable to detect the failing learners while the logistic regression model performs better on the passing group. The Random Forest solution provides good results on balanced courses but clearly fails on very unbalanced ones (see the 28% accuracy on the passing group). The SVM model presents results similar to the logistic regression model, except for the passing learner accuracies which are clearly lower.

Compared to the best shallow models, the dense neural network presents a poor performance on passing learners and does not significantly outperform failing learner prediction. In our opinion, the lack of passing learners for each course in the train set does not enable to fit appropriately the parameter of the network. The LSTM model provides a result similar to the dense neural network, but with a higher computation cost. Hence we do not recommend these neural-based models in our context.

The process mining model presents very low scores. Our explanation is that the graphs generated by the heuristic miner algorithm on the two learner groups (passing and failing) contain the same navigation paths. Traces for passing or failing learners can then be replayed on both graphs with a good fitness.

A similarly bad result is also obtained with the sequence mining model. It can be explained by the fact that frequent sequences of failing learners are short and almost all included in the frequent sequences of the passing learners. Passing frequent patterns are more numerous and longer (longer patterns involve an increase of the similarity score between a model and an input learner trace). It is then more likely to find a better similarity between a learner sequence and the passing learner model. Our conclusion for the process and sequence mining approaches is that passing and failing learners do not present a discriminant behavior on the way they browse the courses.

The second and third parts of table 4 present the average accuracies when temporal features are used for the input dataset of the models, without and with stacking. Compared to the results obtained with the basic features, the results with the temporal features with and without stacking are lower for all models on all sequence lengths, with a clear drop on the accuracies of the passing learners. To conclude, the temporal features do not provide any improvement in our experiments.

To summarize, our experiments show that shallow models present the best results for our dataset. Among them, AdaBoost and logistic regression present the best results respectively for the failing group and the passing group. Another observation is that contrary to several experimental results [8, 9], our temporal data mining approaches (temporal features, LSTM, process mining and sequence mining solutions) do not perform well on our data set. Our conclusion here is that no difference can be found in the way learners access the course resources.

7. CONCLUSION

The objective of our work was to assess several solutions for predicting success in the context of Massive Online Open Courses, using a new dataset provided by the OpenClassrooms company, a major online course enterprise in France.

From our experimental results, we reached the following conclusions:

- failing and passing learners do not seem to present differences in the way they browse a course. Neither specific paths nor specific patterns are identified with our proposed solutions to discriminate between passing and failing learners;
- best features depend on the model used for the prediction tasks;
- temporal features do not increase the performance of the prediction task;
- the best models to detect failing and passing learners are respectively based on AdaBoost and logistic regression solutions.

A short term perspective work is to apply the same prediction tasks on other MOOC datasets, in order to validate our previous conclusions in other learning environments.

8. REFERENCES

- [1] J. Gardner and C. Brooks, "Student success prediction in MOOCs," *User Modeling and User-Adapted Interaction*, vol. 28, pp. 127–203, June 2018.
- [2] C. Taylor, K. Veeramachaneni, and U. O'Reilly, "Likely to stop? predicting stopout in massive open online courses," *CoRR*, vol. abs/1408.3382, 2014.
- [3] I. Koprinska, J. Stretton, and K. Yacef, "Students at risk: Detection and remediation," in *Proceedings of the 8th International Conference on Educational Data Mining, EDM*, pp. 512–515, Jun. 2015.
- [4] J. Feild, "Improving student performance using nudge analytics," in *Proceedings of the 8th International Conference on Educational Data Mining, EDM*, pp. 464–467, Jun. 2015.
- [5] S. Tomkins, A. Ramesh, and L. Getoor, "Predicting post-test performance from online student behavior: A high school MOOC case study," in *Proceedings of the 9th International Conference on Educational Data Mining, EDM*, pp. 239–246, Jun. 2016.
- [6] M. Sweeney, H. Rangwala, J. Lester, and A. Johri, "Next-term student performance prediction: A recommender systems approach," *Journal of Educational Data Mining (JEDM)*, vol. 8, no. 1, 2016.
- [7] Z. Ren, H. Rangwala, and A. Johri, "Predicting performance on MOOC assessments using multi-regression models," in *Proceedings of the 9th International Conference on Educational Data Mining, EDM*, pp. 484–489, Jun. 2016.
- [8] K. H. R. Ng, K. Hartman, K. Liu, and A. W. H. Khong, "Modelling the way: Using action sequence archetypes to differentiate learning pathways from learning outcomes," in *Proceedings of the 9th International Conference on Educational Data Mining, EDM*, pp. 167–174, Jun. 2016.
- [9] P. Mukala, J. C. Buijs, M. Leemans, and W. M. P. van der Aalst, "Learning analytics on coursera event data: A process mining approach," in *SIMPDA (P. Ceravolo and S. Rinderle-Ma, eds.)*, vol. 1527 of *CEUR Workshop Proceedings*, pp. 18–32, 2015.
- [10] M. S. Boroujeni, K. Sharma, L. Kidzinski, L. Lucignano, and P. Dillenbourg, "How to quantify student's regularity?," *Proceedings of the 11th European Conference on Technology Enhanced Learning*, pp. 15. 277–291, 2016.
- [11] F. Wang and L. Chen, "A nonlinear state space model for identifying at-risk students in open online courses," in *Proceedings of the 9th Intl Conference on Educational Data Mining*, pp. 527–532, Jun. 2016.
- [12] M. Sadallah, B. Encelle, A. E. Maredj, and Y. Prié, "Towards reading session-based indicators in educational reading analytics," in *Proceedings of the 10th European Conference on Technology Enhanced Learning, EC-TEL, Toledo, Spain, September 15-18*, pp. 297–310, 2015.
- [13] J. Whitehill, J. J. Williams, G. Lopez, C. A. Coleman, and J. Reich, "Beyond prediction: First steps toward automatic intervention in mooc student stopout," in *Proceedings of the 8th International Conference on Educational Data Mining, EDM*, pp. 171–196, 2015.
- [14] Y. Chen, Q. Chen, M. Zhao, S. Boyer, K. Veeramachaneni, and H. Qu, "Dropoutseer: Visualizing learning patterns in massive open online courses for dropout reasoning and prediction," in *Conference on Visual Analytics Science and Technology, VAST*, pp. 111–120, Oct. 2016.
- [15] S. Nagrecha, J. Z. Dillon, and N. V. Chawla, "Mooc dropout prediction: Lessons learned from making pipelines interpretable," in *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 351–359, Apr. 2017.
- [16] W. M. P. V. der Aalst, *Process Mining: Data Science in Action*. Heidelberg: Springer-Verlag Berlin Heidelberg, 2nd ed., 2016.

Scholars Walk: A Markov Chain Framework for Course Recommendation

Agoritsa Polyzou
University of Minnesota
polyz001@umn.edu

Athanasios N.
Nikolakopoulos
University of Minnesota
anikolak@umn.edu

George Karypis
University of Minnesota
karypis@umn.edu

ABSTRACT

Course selection is a crucial and challenging problem that students have to face while navigating through an undergraduate degree program. The decisions they make shape their future in ways that they cannot conceive in advance. Available departmental sample degree plans are not personalized for each student, and personal discussion time with an academic advisor is usually limited. Data-driven methods supporting decision making have gained importance to empower student choices and scale advice to large cohorts. We propose *Scholars Walk*, a random-walk-based approach that captures the sequential relationships between the different courses. Based on the “wisdom of the crowd” and the students’ prior courses, we recommend a short list of courses for next semester. Our experimental evaluation illustrates that Scholars Walk outperforms other collaborative filtering and popularity-based approaches. At the same time, our framework is very efficient, easily interpretable, while also being able to take into consideration important aspects of the educational domain.

Keywords

course recommendation, Markov chains, random walks, sequential recommendation, higher education

1. INTRODUCTION

The general purpose of higher education is to offer programs, which will help learners to gain knowledge throughout their studies. Students enjoy a plethora of offerings. However, course selection can be “messy and unorganized” [3] as it depends on many factors that students need to consider. Students have to balance personal preferences (interests, objectives, and career goals) and general education and degree program requirements. As a result, course selection can be a non-trivial task.

Decisions can be made based on manual guides offered from each department, but these are not tailored to individual

cases [7] in a higher education setting. Personalized assistance can be given by academic advisers, however this is not scalable with large cohorts with thousands of students. The ratio of student to advisor may be very high [14], limiting the adviser-advisee discussion time. Additionally, college students take on average up to 20% more courses than required [2]. Better advising can help alleviate these problems. We need predictive models that can be employed to enable strategic action and attain better results. In this paper, we focus on appropriately designing a course recommendation system (CRS) that could facilitate the conversation between advisors and students for future planning.

There are several existing approaches to generate a set of courses to recommend for next semester. Their majority suggest courses based on either the constraints and requirements that they satisfy or their expected grades. This paper introduces *Scholars Walk*, a random-walk based approach for the course selection problem. It describes a personalized model that takes advantage of the sequential nature of course selection. We assume that students’ choices for the next term depend on the courses they have taken so far. In our approach, we build a Markov chain for each degree program over the courses taken consecutively. Then, we perform a random walk, starting from the courses that students took in the previous semester. We evaluated the proposed approach on a number of different departments with different subjects and characteristics. Scholars Walk overall outperforms other competing approaches in all the metrics considered in this paper.

2. RELATED WORK

Recommender systems have been broadly applied within the context of student learning [16]. We will further review the different approaches developed to help students select a subset of courses to register for an upcoming semester. The first course recommender systems are based on constraint satisfaction [22]. The sequence-based recommender [24] also considers complex constraints to improve the expected time-to-degree and GPA. A related body of work involves mining of association rules. Al-Badarenah et al. [1] cluster the students based on their grades first. Nguyen et al. [18] apply sequential rule mining in (course, grade) pairs and recommend the courses with the best performance. A different CSR was proposed by Esteban et al. [10], where there is available information about students’ satisfaction after taking a course.

Agoritsa Polyzou, Athanasios N. Nikolakopoulos and George Karypis "Scholars Walk: A Markov Chain Framework for Course Recommendation" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 396 - 401

Table 1: Statistics for each major.

Major	n	m	grades	%pop	flex
Accounting	846	53	22,524	45.9	0.28
Aerospace Engr	532	109	16,259	25.7	0.10
Biology	1,275	146	28,084	14.9	0.11
Biol Society Env	709	57	14,597	31.4	0.31
Biomedical Engr	644	131	19,748	23.8	0.16
Chemical Engr	826	108	24,825	26.3	0.11
Chemistry	724	145	18,292	17.4	0.14
Civil Engr	651	112	19,189	26.5	0.12
Communication	1,333	95	22,421	15.4	0.19
Computer Sc	998	161	24,899	13.7	0.11
Electrical Engr	740	164	22,191	17.1	0.12
Elementary Ed	770	49	16,527	40.4	0.31
English	1,176	153	17,736	9.5	0.11
Finance	1,234	83	32,255	29.7	0.20
Genetics Cell	680	93	15,385	23.1	0.19
Journalism	2,306	100	40,519	17.1	0.20
Kinesiology	1,176	164	33,622	14.8	0.16
Marketing	1,291	69	29,901	30.8	0.20
Mechanical Engr	1,369	132	39,436	18.9	0.11
Nursing	819	86	25,136	31.2	0.27
Nutrition	554	87	15,591	29.7	0.19
Political Science	1,307	171	19,260	8.1	0.12
Psychology	1,894	115	31,141	13.4	0.15

n , m are the number of students and courses.

%pop is the course popularity (percentage of students that took a course at least once).

The last column (flex) is the degree flexibility.

Recently, recurrent neural networks (RNNs) have been successfully applied within the educational domain. Long Short Term Memory (LSTM) networks have been used for grading prediction [13, 20]. In terms of course recommendation, a combination of LSTMs and skip-gram models has also developed to balance implicit and explicit student preferences [23]. Morsy et al. [17] have also used RNN to recommend courses which are expected to help maintain or improve students' GPA. Other approaches include a Markov-based model [15], that represents the sequence of courses taken as a stochastic process. Garner et al. [11] build a co-enrollment network and extract features for a network-based structural model. Finally Elbadrawy et al. [9] propose using the academic features to improve the recommendation performance.

3. DOMAIN & DATASET

This work focuses on the undergraduate students in a traditional educational institution. We used a dataset from the University of Minnesota that spans more than 10 years. The A–F grading scale (A, A–, B+, B, B–, C+, C, C–, D+, D, F) is used. Courses in which a student receives less than a C– do not count toward satisfying degree requirements.

We extracted the degree programs that have at least 500 graduated students from 23 different majors. We only kept students that actually received their degree and had at least three consecutive semesters with valid courses. We selected the 40 most frequent courses and the courses that belonged to frequent subjects. A subject is considered frequent if stu-

dents have taken at least three courses that belong to that subject on average. We removed instances without an A–F grade, and non-academic courses, like independent/directed study or field study. We did not consider offerings in the summer semester. As these are less common, they would distort the course sequence of students not enrolled in summer. Basic statistics for each degree program are shown in Table 1. The average course popularity (%pop) for course i is the percentage of students that have taken i at least once during their studies. The degree flexibility (flex) is a measure of how different are the course selections that students make. It is one minus the average Jaccard similarity coefficient for every pair of students. The Jaccard similarity is computed as the number of courses that two students have in common divided by the minimum courses that student has taken them.

4. PROPOSED METHOD

4.1 Assumptions & Notation

In the context of course recommendation for higher education, we make the following assumptions:

1. Time is discrete and moves in steps, from one semester to the next.
2. There is a relative ordering of the courses in terms of course levels, difficulty or material covered.
3. Learning is seldom non-sequential; each course completed provides some knowledge and experience that can be used in future courses. As a consequence, sequence matters in course selection.
4. In the absence of enough domain experts, the order in which courses are taken by students historically can reveal useful information on the curriculum and degree requirements.
5. We know the number of courses that the student will take next semester.

For the rest of the paper we will adopt the following notation. When we use the word *target* we will refer to the student/course/semester for which we want to generate recommendation. Matrices are denoted with capital bold letters, while vectors are denoted with lower bold letters. Calligraphic letters will be used for sets.

The set of students is \mathcal{S} and has size m . The set of all courses is denoted by \mathcal{C} , $|\mathcal{C}| = n$. Student j has an enrollment history \mathcal{H}_j , that is an ordered set of courses, $\{\mathcal{C}_{j,1}, \dots, \mathcal{C}_{j,t}, \dots, \mathcal{C}_{j,t_j}\}$, where $\mathcal{C}_{j,t}$ is the set of courses taken in semester t and t_j is the last semester that the student took courses. Table 2 presents the symbols we used.

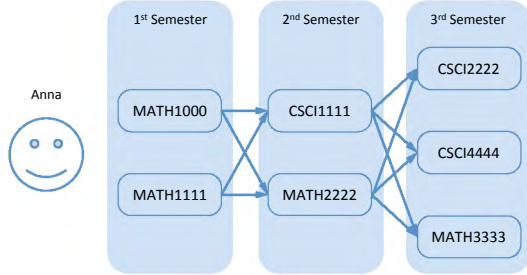
4.2 Building the Markov chain

Markov models satisfy the Markov property, i.e., the conditional probability distribution of future states depends only on the current state. In the simplest Markov model, known as first-order, each state is formed by a single action, i.e., a student took a course. In the case of K -th-order models, the state-space will correspond to all possible sequences of K actions. As the available data could not adequately support the number of states of higher-order chains, these models would suffer from reduced coverage and possibly worse overall performance [6]. Therefore, we adopted a first-order

Table 2: Notation.

n, m	number of courses, students
i, i'	indexes for courses
j, j'	indexes for students
t_j	number of semesters that j has taken courses
t	index for semesters
\mathcal{C}, \mathcal{S}	set of courses, courses
\mathcal{A}	set of states in Markov chain $\{\mathcal{A}_1, \dots, \mathcal{A}_n\}$
\mathcal{H}_j	enrollment history of student j
$\mathcal{C}_{j,t}$	course that student j took in semester t
\mathbf{T}, \mathbf{F}	matrices ($n \times n$)
$\mathbf{T}_{k,l}$	the (k, l) element of matrix \mathbf{T}
\mathbf{T}_k	the k -th row of matrix \mathbf{T} ($1 \times n$)
$\mathbf{T}_{:,l}$	the l -th column of matrix \mathbf{T} ($n \times 1$)
\mathbf{u}	personalization vector ($1 \times n$)
$\mathbf{p}^{(k)}$	state vector ($1 \times n$) at timestep k

Figure 1: Example: Anna’s enrollment history.



Markov chain. We assume that the next-semester courses depend only on the courses that the student is taking the current semester.

Markov models are represented by the parameters $(\mathcal{A}, \mathbf{T})$, where \mathcal{A} is the set of states for which the Markov model is defined; and \mathbf{T} is an $(n \times n)$ transition probability matrix (TPM), where n is the number of states (i.e., courses). In this context, state \mathcal{A}_i is associated with the fact that the student took the course i . Each entry $\mathbf{T}_{i,i'}$ corresponds to the probability of moving to state $\mathcal{A}_{i'}$ when the process is in state \mathcal{A}_i , i.e., taking course i' after course i . Note that this matrix is not symmetric, i.e., $\mathbf{T}_{i,i'} \neq \mathbf{T}_{i',i}$, as the order in which the courses are taken matters.

Based on the historical enrollment information of the students, we first compute \mathbf{F} , an $(n \times n)$ matrix that holds the counts of every pair of consecutive courses. Every pair of courses (i, i') that a student has taken consecutively is used to estimate the entry $\mathbf{F}_{i',i}$, i.e., the frequency of the event that state $\mathcal{A}_{i'}$ follows the state \mathcal{A}_i . For example, consider student Anna in Fig. 1. The entry corresponding to the course pair of (MATH1000, CSCI1111) will be updated. Similarly, every line connecting two courses will equally contribute in the corresponding element of matrix \mathbf{F} .

After we compute the frequencies of matrix \mathbf{F} , we need to normalize it to get \mathbf{T} , a row stochastic matrix, so that the total transition probability from state i to any other state

will sum up to 1:

$$\mathbf{T}_i = \mathbf{F}_i / \sum_{i'=1}^n \mathbf{F}_{i,i'}, \text{ if } \sum_{i'=1}^n \mathbf{F}_{i,i'} > 0.$$

Additionally, it is possible that the sum of some rows to be zero. This occurs when a course is taken at the last semester of every student, so there are no courses after that to pair it with. In that case, we set the diagonal elements of the zero rows to one; $\mathbf{T}_{i,i} = 1$ and $\mathbf{T}_{i,i'} = 0$ for $i \neq i'$, if $\sum_{i'=0}^n \mathbf{F}_{i,i'} = 0$.

4.3 Walking over courses

We can view the Markov chain in the context of random walk on a course-to-course graph that is governed by the transition probability matrix. A random walk on a directed graph will form a path of vertices generated from a start vertex by selecting an edge, making a step by traversing the edge to a new vertex, and repeating the process [4]. This concept has been applied to many scientific fields. Closer to this work, random walks have recently been used for top-n item recommendation [19], and they are also known to empower systems used in production at major social media platforms [12, 8].

A random walk starts with any probability distribution $\mathbf{u} \in \mathcal{R}^{1 \times n}$. \mathbf{u}_i is the probability of starting at vertex i . If one starts at a vertex i , then $\mathbf{u}_i = 1$, else $\mathbf{u}_{i'} = 0$ for $i' \neq i$. In our setting, the random walk for student j will equally start from any course in the student’s last semester, so the personalization vector will be:

$$\mathbf{u}_i = \begin{cases} 1/|\mathcal{C}_{j,t_j}| & \text{if } i \in \mathcal{C}_{j,t_j}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Let $\mathbf{p}^t \in \mathcal{R}^{1 \times n}$ be a row vector with an element for each vertex specifying the probability of being there at time t . Before we start the walk, $\mathbf{p}^0 = \mathbf{u}$. After the first step, the probability of being at vertex i' is the sum over each adjacent vertex i of starting at i and taking the transition from i to i' . In matrix notation, when we are at state k and we take a step, we will get the following probability distribution:

$$\mathbf{p}^{k+1} = \mathbf{p}^k \mathbf{T}, \quad (2)$$

where the i -th entry of the \mathbf{p}^{k+1} is the probability of the walk after $k+1$ steps to land at vertex i . This can be written as a function of the starting probability vector as:

$$\mathbf{p}^{k+1} = \mathbf{u} \mathbf{T}^k. \quad (3)$$

The probability of the walker to reach the vertices after K steps provides an intuitive measure that can be used to rank the courses and offer personalized recommendations to the student accordingly.

Scholars Walk

To introduce an additional way for personalization in our model, we perform a *random walk with restarts* [21]. We introduce a parameter α , $0 < \alpha \leq 1$ that controls if the walk will take the step described above, or if the walk will restart. In the latter case, we use the personalized probability distribution as the restarting distribution. The probability dis-

Algorithm 1 SCHOLARS WALK

Input: Model \mathbf{T} , student's personalization vector \mathbf{u} , parameters α, β , number of steps K .
Output: Recommendation vector \mathbf{p}^{rec} .
 $\mathbf{p}^0 \leftarrow \mathbf{u}, k \leftarrow 0$
repeat
 $k \leftarrow k + 1$
 $\mathbf{p}^k \leftarrow \alpha \mathbf{p}^{k-1} \mathbf{T} + (1 - \alpha) \mathbf{u}$ \triangleright Take a step.
 $\mathbf{p}^k \leftarrow \mathbf{p}^k / \|\mathbf{p}^k\|_1$ \triangleright Normalize \mathbf{p}^k .
until $\|\mathbf{p}^k - \mathbf{p}^{k-1}\|_2 < tol$ or $k \geq K$
for $i \leftarrow 1$ to n **do**
 $\mathbf{p}_i^k \leftarrow \mathbf{p}_i^k * \text{pop}_i^{-\beta}$ \triangleright Penalize popular courses.
end for
 $\mathbf{p}^{rec} \leftarrow \mathbf{p}^k$

tribution now is defined as:

$$\begin{aligned} \mathbf{p}^{k+1} &= \alpha \mathbf{p}^k \mathbf{T} + (1 - \alpha) \mathbf{u} \\ &= \mathbf{p}^k (\alpha \mathbf{T} + (1 - \alpha) \mathbf{1} \mathbf{u}) \\ &= \mathbf{u} (\alpha \mathbf{T} + (1 - \alpha) \mathbf{1} \mathbf{u})^k, \end{aligned} \quad (4)$$

where $\mathbf{1}$ is a column vector ($n \times 1$) of ones. The product of $\mathbf{1} \mathbf{u}$ will give us an $(n \times n)$ matrix where every row will have the probability that the walk will start at the corresponding course. Scholars Walk will perform a random walk governed by the matrix $\alpha \mathbf{T} + (1 - \alpha) \mathbf{1} \mathbf{u}$.

The exact steps we followed are shown in Alg. 1. We can specify the number of steps to perform, or we can allow the algorithm to converge. If the number of steps is very small, the walk might not explore enough courses. If the number of steps is large, the walk might travel too far, and the recommendations might not be so relevant for the student. Additionally, to limit the domination of popular courses, we penalize the probabilities with the term $\text{pop}_i^{-\beta}$ [5], where pop_i is the popularity of the course. The parameter $\beta, 0 < \beta \leq 1$ shows how harsh we need to be with the penalty term.

Scholars Walk allows us to consider direct, as well as, transitive relations between the courses. It also provides a considerable degree of personalization, in order to recommend courses that are relevant to each particular student.

5. EXPERIMENTAL DESIGN

5.1 Competing approaches

The baselines are two group popularity approaches, on the department level (**Pop1**) and the academic level (**Pop2**) of the student measured by the number of years in the program [9]. For Pop1, we recommend the most popular courses in the major. For Pop2, we recommend the most common courses on the major and the academic level of the student ("freshmen", "sophomores", "juniors", and "seniors"). Students after their fourth year are considered seniors.

We also compared against Basic Markov model (**Markov**) and Basic Markov model with skip (**MarkovSkip**) [15]. In these models, for a target student, the set of courses that other students have taken after taking a course that the target student took are the possible courses to recommend. We consider the combination of courses during the last two

semesters to build and test the model. Each course is assigned a recommendation score that is the sum of all the conditional probabilities that lead to that course starting from the student's enrollment in the last semester. While the counts used in this case are the same with the ones computed in our matrix \mathbf{F} , the conditional probabilities are computed differently. In order to produce recommendations for students whose set of prior courses did not have a match, the skip model was introduced. In that case, we find other students that have similar course history with the target student, and weight their corresponding probabilities by a parameter λ .

Last, we train an **LSTM**-based course prediction model similar to [17, 23]. LSTMs can learn temporal dependencies with additional gates to retain and forget selected information. As input, we use a multi-hot representation of course enrollments per semester which are mapped to a predicted sequence of vectors. Once the LSTM has been learnt, we feed the network with a binary vector that indicates the courses that the target student has taken the past semester. The weights at the output of the model are used to rank the courses.

5.2 Evaluation metrics

Like in prior work [9, 15, 17, 23], we used **Recall@ n_s** as the primary evaluation metric for the predictions, where n_s is the number of courses that the student took in the target semester. This is the percentage of actual enrolled courses that were contained in the recommendation list. The reported metrics are averaged out across all students predicted. Note that recall and precision are equivalent in our setting, since we recommend exactly as many courses as the student will take the upcoming semester.

We also compute the percentage of queries for which we were able to retrieve at least one of the courses that the student took in the target semester (**%rel**). It measures for how many cases we were able to recommend at least one course that was relevant.

5.3 Experimental setting

Model selection. Using the dataset described in Sect. 3, we split it into train, validation and test sets as follows. All semesters before 2013 (about 10 years) were used for training, courses taken during 2013 and in Spring 2014 were used for validation, and courses taken afterwards (Fall 2014 to Spring 2017) were used for test purposes, to report the results. The training set was used for building the models, whereas the validation set was used to select the best performing parameters in terms of the highest Recall@ n_s . Based on the best set of parameters for the validation set, we computed the test set results in Sect. 6.

Parameters. For parameter α , we tried the following set of values: {1e-4, 1e-3, 1e-2, 1e-1, 0.2, 0.4, 0.6, 0.7, 0.8, 0.85, 0.9, 0.99, 0.999}. For parameter β , we tested values from 0 to 0.8, in increments of 0.025. In terms of the number of steps that we allowed for our walker, we tested the values 1, 3, and 1000. The last value corresponds to no limit for the number of steps.

Additional filtering. We build a different model for each

Table 3: Results for Scholars Walk w.r.t. K .

K	Recall@ n_s	%rel	α	β	avg#steps
1	0.466	75.1	0.955	0.047	1
3	0.460	74.6	0.088	0.053	1.95
1000	0.461	74.6	0.075	0.051	2.32

K is the number of steps that we allow to our walker. α, β columns show the average values of these parameters over the models of all the majors.

The last column shows the actual average number of steps the Scholars Walk made before convergence.

Table 4: Performance comparison.

Model	Recall@ n_s	%rel
Pop1	0.336	62.5
Pop2	0.338	64.6
Markov	0.456	73.0
MarkovSkip	0.400	69.6
LSTM	0.406	69.6
Scholars Walk	0.466	75.1

major for all the approaches we tested. After we generate a ranked list of the courses using any method, we filter out courses that are not offered the target semester. We also remove courses that the student has taken in the past and achieved a grade above C-, as they do not count towards any degree requirements, as mentioned in Sect. 3. In the end, we return a list with as many recommendations as the number of courses, n_s , that the student took next semester, based on assumption 5.

6. RESULTS

In this section, we will try to answer the following questions: 1) How do the parameters in our models affect the overall performance? Specifically, how does the number of steps affect recommendation performance? 2) What is the performance of our approach compared to the state-of-the-art approaches?

6.1 The effect of the number of steps

The performance of our models in terms of the metrics computed for different values of K is shown in 3. For each model and selection of K , we see the values of the parameters α and β that were used. These parameters were selected based on the recall on the validation set. The parameter α controls the restarting probabilities, while β is used to re-weight the probability distribution before recommending its highest-weighted courses. The column avg#steps shows the average number of steps that the Scholars Walk actually made before convergence.

In this domain, we need only a few steps, as we can understand from Table 3: not only when we set $K = 1$ we get the best performance, but also, when we allow the walk to take many steps, the parameter α gets smaller values. This forces the walk to go back to the student’s personalized starting vector with higher probability, indicating that the starting distribution is very important. Additionally, even if we do not put any constraints in K , the number of steps that the

Scholars Walk takes is quite small. There is a small increase when increasing K from 1 to 3, but after that, the number of steps actually taken is not that high.

It is worth pointing out that, while setting $K = 1$ gives us the best overall performance, this is not the case for all the departments. The right value for K depends on the dataset used. In our data, there are four departments that need these extra steps. We observed that these departments have low average course popularity, which is average percentage of students that have taken a course at least once at some point during their studies, over all the courses. The average value for the departments with $K > 1$ was $16.7 \pm 9.7\%$, while for the rest of the models the corresponding number is $24.1 \pm 7.2\%$. A stronger signal is present in the metric of the degree flexibility, which is the average Jaccard distance between the courses that any pair of students took, as defined in the end of Sect. 3. The departments with $K > 1$ have 0.118 ± 0.005 degree flexibility against 0.184 ± 0.066 of the rest of the departments. This is an indicator that for stricter degrees, the walk depends on the extra steps to explore more courses. In these departments, students will take overall very similar sets of courses. On the other hand, if the degree program offers more freedom to the students, they select a wider range of courses, and there are more connections within courses.

6.2 Performance comparison

By comparing the best Scholars Walk model against five competing approaches, we get the results on Table 4. Our model performs the best, both in terms of recall, and in the percentage of cases for which it manages to be return some relevant recommendations.

Popularity approaches are having considerably satisfactory performance. However, specifying the academic level of the student does not help much. They can recommend relevant courses to more than 60% of the cases. The two Basic Markov models have quite different performance. The Markov model with skips performs poorly, compared to the Basic model. Additionally, it is worth mentioning that the Skip model was performing better and better as the parameter λ was getting smaller. The weight of the cases that do not completely match the target student’s history, have as weight a power of λ . Consequently, when $\lambda \rightarrow 0$, the Skip model becomes the Basic Model. For that reason, the smaller value of λ that we report results for, is 0.4.

While comparing the Basic Markov model with Scholars Walk, it may seem that they have similar performance. However, that might be misleading, as the Basic Markov model utilizes longer course enrollment history than the Scholars Walk. It looks back two semesters on the student’s courses, which corresponds to a second-order Markov chain. Moreover, the model uses data from two semesters not only for computing the associated probabilities, but also to make predictions. This leads to increased complexity because of the larger state-space with no benefit in recommendation quality. In the same boat are the LSTMs as well. Their increased complexity might lead to the overfitting of the model, when the data are not sufficient for training. Our approach, which is a first-order Markov chain, manages to perform better than the higher-order models and LSTMs.

Scholars Walk can accurately predict the course selection of the students, by taking advantage of the “breadth and depth” of the data. In terms of time complexity, once we build the transition probability matrix, walking through the courses is trivial. As a result, it scales well with the number of students, while providing them personalized recommendations. At the same time, it is a white-box model, where the recommendations are easily explainable.

7. CONCLUSION

In this paper we propose Scholars Walk, a novel method designed to harvest the sequential patterns arising from past course enrollment data in order to recommend a short list of personalized course suggestions for the next semester. The proposed method relies on a random walk-based scheme on a course-to-course graph and personalization is achieved by a student-adapted starting distribution reflecting the current student’s enrollments. When compared with five competing models, from popularity-based to LSTMs and Basic Markov models, Scholars Walk achieves the best performance. It manages to be a successful, scalable approach that provides personalized recommendations for every student.

8. ACKNOWLEDGMENTS

This work was supported in part by NSF (1447788, 1704074, 1757916, 1834251), Army Research Office (W911NF1810344), Intel Corp, and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute.

9. REFERENCES

- [1] A. Al-Badarenah and J. Alsakran. An automated recommender system for course selection. *Intl. Journal of Advanced Computer Science and Applications*, 7(3):1166–1175, 2016.
- [2] C. C. America. Time is the enemy, 2011.
- [3] E. Babad and A. Tayeb. Experimental analysis of students’ course selection. *British Journal of Educational Psychology*, 73(3):373–393, 2003.
- [4] A. Blum, J. Hopcroft, and R. Kannan. Random walks and markov chains. In *Foundations of data science*. Vorabversion eines Lehrbuchs, 2016.
- [5] F. Christoffel, B. Paudel, C. Newell, and A. Bernstein. Blockbusters and wallflowers: Accurate, diverse, and scalable recommendations with random walks. In *9th ACM Conf. on Recommender Systems*, pages 163–170, New York, NY, USA, 2015. ACM.
- [6] M. Deshpande and G. Karypis. Selective markov models for predicting web page accesses. *ACM Trans. on Internet technology (TOIT)*, 4(2):163–184, 2004.
- [7] A. Diamond, J. Roberts, T. Vorley, G. Birkin, J. Evans, J. Sheen, and T. Nathwani. Uk review of the provision of information about higher education: advisory study and literature review: report to the uk higher education funding bodies by cfe research. 2014.
- [8] C. Eksombatchai, P. Jindal, J. Z. Liu, Y. Liu, R. Sharma, C. Sugnet, M. Ulrich, and J. Leskovec. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *World Wide Web Conf.*, pages 1775–1784, 2018.
- [9] A. Elbadrawy and G. Karypis. Domain-aware grade prediction and top-n course recommendation. In *10th ACM Conf. on RecSys*, pages 183–190, 2016.
- [10] A. Esteban, A. Z. Gómez, and C. Romero. A hybrid multi-criteria approach using a genetic algorithm for recommending courses to university students. In *11th Intl. Conf. on Educational Data Mining*, 2018.
- [11] J. P. Gardner, C. Brooks, and W. Li. Learn from your (markov) neighbor: Coenrollment, assortativity, and grade prediction in undergraduate courses. *Journal of Learning Analytics*, 5(3):42–59, 2018.
- [12] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. Wtf: The who to follow service at twitter. In *22Nd Intl. Conf. on World Wide Web*, pages 505–514, New York, NY, USA, 2013. ACM.
- [13] Q. Hu and H. Rangwala. Course-specific markovian models for grade prediction. In *Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, pages 29–41. Springer, 2018.
- [14] A. Kadlec, J. Immerwahr, and J. Gupta. Guided pathways to student success perspectives from indian college students and advisors. *New York: Public Agenda*, 2014.
- [15] E. S. Khorasani, Z. Zhenge, and J. Champaign. A markov chain collaborative filtering model for course enrollment recommendations. In *Big Data (Big Data), IEEE Intl. Conf. on*, pages 3484–3490. IEEE, 2016.
- [16] N. Manouselis, H. Drachsler, R. Vuorikari, H. Hummel, and R. Koper. Recommender systems in technology enhanced learning. In *Recommender systems handbook*, pages 387–415. Springer, 2011.
- [17] S. Morsy and G. Karypis. Learning course sequencing for course recommendation. 2018.
- [18] H.-Q. Nguyen, T.-T. Pham, V. Vo, B. Vo, and T.-T. Quan. The predictive modeling for learning student results based on sequential rules. *Intl. Journal of Innovative Computing, Information and Control (IJICIC)*, 14(6):2129–2140, 2018.
- [19] A. N. Nikolakopoulos and G. Karypis. Recwalk: Nearly uncoupled random walks for top-n recommendation. In *12th ACM Intl. Conf. on Web Search and Data Mining*, pages 150–158. ACM, 2019.
- [20] F. Okubo, T. Yamashita, A. Shimada, and H. Ogata. A neural network approach for students’ performance prediction. In *Seventh Intl. Learning Analytics & Knowledge Conf.*, pages 598–599. ACM, 2017.
- [21] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
- [22] A. Parameswaran, P. Venetis, and H. Garcia-Molina. Recommendation systems with complex constraints: A course recommendation perspective. *ACM Trans. on Information Systems (TOIS)*, 29(4):20, 2011.
- [23] Z. A. Pardos, Z. Fan, and W. Jiang. Connectionist recommendation in the wild: on the utility and scrutability of neural networks for personalized course guidance. *User Modeling and User-Adapted Interaction*, pages 1–39, 2019.
- [24] C. Wong. Sequence based course recommender for personalized curriculum planning. In *Intl. Conf. on Artificial Intelligence in Education*, 2018.

A Comparison of Automated Scale Short Form Selection Strategies

Anthony W. Raborn
Pasco County Schools
7227 Land O' Lakes Blvd.
Land O' Lakes, FL 34638
1-727-774-4233
araborn@pasco.k12.fl.us

Walter L. Leite
University of Florida
1215 Norman Hall
Gainesville, FL, 32611
1-352-273-4302
Walter.leite@coe.ufl.edu

Katerina M. Marcoulides
University of Florida
1215 Norman Hall
Gainesville, FL, 32611
1-352-273-4332
k.marcoulides@coe.ufl.edu

ABSTRACT

Short forms of psychometric scales have been commonly used in educational and psychological research to reduce the burden of test administration. However, it is challenging to select items for a short form that preserve the validity and reliability of the scores of the original scale. This paper presents and evaluates multiple automated methods for scale short form creation based on metaheuristic optimization algorithms that incorporate validity criteria based on internal structure and relationships with other variables. The ant colony optimization (ACO) algorithm, tabu search (TS), simulated annealing (SA) and genetic algorithm (GA) are examined using confirmatory factor analysis (CFA) of scales with one factor, three factor, and bi-factor factorial structure. The results indicate that SA created short forms with best model fit for scales with one and three factor structures, but ACO was able to obtain highest reliability. For scales with bi-factor structure, SA provide short forms with best model fit, but TS obtained highest reliability. Overall, the SA algorithm is recommended because it produced consistently best model fit and reliability that was only slightly lower than the ACO or TS algorithms.

Keywords

Short form development, confirmatory factor analysis, metaheuristic algorithms, validity, ant colony optimization, tabu search, genetic algorithm, simulated annealing

1. INTRODUCTION

Applied researchers using psychometric scales often face a dilemma due to limited resources: should they use the full form of a well-established scale with strong validity evidence supporting it, but with a large number of items requiring a substantial amount of time and effort to complete, or should they use a short form of the scale that has not had the extensive evidence of validity? This issue has generated strong interest in the academic community the development of short forms of scales (e.g. [1]). Multiple methods have been proposed for scale short form development [2], with different fields utilizing a few preferred methods. For example, these methods include theoretical or practical justifications for the

inclusion or exclusion of items (e.g., [3]), keeping one item from a set of items that are apparently similar or redundant (e.g., [4]), obtaining certain criteria for statistical values such as high factor loadings or item correlations (e.g., [5]), adding or retaining items that seem to improve measures of reliability and/or dimensionality (e.g., [6]).

The focus of item selection for short forms tends to be on the internal structure of the newly-created form, rather than using external relationships to help build the short form. For example, Petrillo, Capone, Caso, and Keyes [7] created a short form for a positive mental health assessment for use with Italian respondents by selecting items from twelve other scales with a focus on its internal structure. The resultant short form had adequate psychometric properties, but the average absolute correlation between the total score and sixteen other criterion measures was 0.37 (range: 0.20 to 0.62). Despite the adequate validity evidence for the internal structure, the external relationships would be characterized as modest since on average the short form's and the other measures' scores shared about 6% of their variances.

Obtaining a short form that has both adequate internal structure and strong validity with respect to relationships with other variables is difficult with traditional methods of short form development. Metaheuristic optimization algorithms [8] have the potential to solve these difficulties because they can simultaneously maximize multiple validity criteria for short forms. This paper aims to present the evaluation of multiple automated methods for short form creation based on metaheuristic optimization algorithms that incorporate criteria based on internal structure and relationships with other variables and determine which perform best under commonly used scale structures.

2. THEORETICAL FRAMEWORK

There have been some attempts to develop algorithms to derive short forms of scales that (a) maintain the internal structure of the scale in question (e.g. factor structure and/or content balance), (b) have favorable model characteristics such as meeting model fit statistic thresholds, and (c) produces scale scores that have favorable relationships with other variables, including other scales or external variables. For example, Olaru, Witthoft, and Wilhelm [9] compared multiple algorithms for the purpose of creating psychometrically valid short forms of a 99-item scale with various criterion (e.g., jointly optimizing two fit indices) and concluded that, under their study conditions, the Ant Colony Optimization (ACO) and Genetic Algorithm (GA) were able to produce statistically appropriate short forms that generalize well to new data. Marcoulides and Drezner [10] have shown that a Tabu search

Anthony Raborn, Walter Leite and Katerina Marcoulides "A Comparison of Automated Scale Short Form Selection Strategies" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 402 - 407

can be used to successfully reduce the number of items loading on factors. Leite, Huang and Marcoulides [2] developed and demonstrated an ACO algorithm that selects items for short forms while keeping adequate model fit and maximizing the relationship between the latent variable and external variables. More recently, Browne, Rockloff, and Rawat [11] produced an automated structural equation modeling (SEM) scale reduction algorithm and purport that it is an effective and efficient method for reducing items during scale development.

While these articles demonstrate the use of some automated scale short form development strategies and the importance of research in this area, an in-depth comparison of automated strategies under different scenarios does not seem to exist. In addition, some commonly-used metaheuristic algorithms for combinatorial problems have never been applied to the short-form development problem. For example, the inaugural example of the simulated annealing (SA) algorithm is with the Traveling Salesman Problem, which has various algorithms attempt to find the shortest path that travels between n cities exactly once [12], while no psychometric use of SA is apparent in the literature. To address these issues, this paper presents a simulation study utilizing three different scale structures commonly observed in educational research (one factor, three factor, and bifactor scales) and four meta-heuristic algorithms: The ant-colony optimization algorithm (ACO), genetic algorithm (GA), Tabu search (TS), and simulated annealing (SA). We chose these algorithms because they are the most well-established metaheuristic algorithms in the combinatorial optimization literature.

The ACO algorithm [13] mimics the behavior of ants searching for the shortest path to a food source. We evaluate the implementation of the ACO algorithm proposed by Leite, Huang and Marcoulides [2] for short form development, with minor modifications to the tuning parameters. Their implementation of the ACO algorithm attaches sampling weights to items, which are used to sample items for a set of candidate short forms of the scales. Each set of candidate short forms is evaluated and the best short form in the set is identified based on criteria that are specified by the researcher, such as SEM fit indices and the relationship between the scale's factors and an external variable. The criteria of choice are used to calculate the pheromone level, which is a summary of the quality of the short form chosen. The pheromone level is then used to update the sampling weights for the next round of sampling of candidate sets of short forms. This is repeated until a specified convergence criterion is met, such as number of iterations without improvement of the solution quality.

The GA mimics the biological process of evolution using the model parameters as genes. As implemented by Yarkoni [14], the algorithm generates an initial population of candidate models of size 200 and, by evaluating the fitness of each candidate model through a loss function, selects the best 20% of the models and repopulates. Between model generations, new models are created from mutation (randomly changing the items in a model) and recombination (two models exchanging items retained). After a certain number of iterations (100+), the model with the best fit according to the loss function is retained as the best solution. The loss function penalizes the fit of the models for every item included; this value needs to be tuned to achieve the correct reduction of items.

The TS algorithm implementation was modified from the presentation given in Marcoulides and Falk [15] to constrain the solution space to a specified number of items. Broadly, the TS looks at each of the local solutions to a model by changing one model

parameter at a time; the particular change can be adjusted to suit the problem at hand. The main idea behind the TS procedure is to continually adjust the currently selected best model by examining other models in the neighborhood of the current best solution. If a neighboring model fits better than the current model, it is selected as the new best fitting model. If not, the examined neighboring model is marked "tabu"—placed on a list so that it is not reconsidered for some number of iterations. For this study, the TS was modified to (a) randomly generate a short form of a predetermined length from a longer form for the first iteration and (b) search for local short forms that maintain the predetermined length.

The SA algorithm is a statistical analog to metallurgic processes of annealing metals [16]. Generally, the algorithm begins with a specified starting model whose parameters are randomly changed by some process and a starting temperature. The new model is compared to the starting model and the difference in model fit is calculated. At any time, if the new model has better fit than the current model, it is selected for use in the next iteration; otherwise, the new model is selected with probability equal to a function of the difference in model fit and the current temperature. After each new model is either selected or ignored, the temperature updates and the current model is randomly changed. The algorithm checks each model against the best model seen and updates as needed. Some variants of the algorithm include a process that selects this best model after a certain number of iterations in which no better model has been found. This process repeats until the temperature reaches zero.

3. METHODS

3.1 Research Questions

1. How do the algorithms differ in terms of the time it takes for each to converge on a short form, model fit and reliability of the short form?
2. How do model misspecifications in the full form affect the fit and reliability of the short forms created by the algorithms?
3. Does the inclusion of an external variable affect the model fit and reliability of the short forms?
4. Does the performance of the algorithms depend of the factorial structure of the scale?

3.2 Manipulated conditions

To investigate the research questions, a Monte Carlo simulation study was conducted using the following population confirmatory factor analysis models: (a) the 20-item unidimensional model of the self-deceptive enhancement (SDE) scale [17], (b) the 24-item three-dimensional model of the teacher efficacy scale [18], and (c) a three-factor bifactor model [20] of the 30-item BASC-2 BESS [19] scale. These models represent three common models seen in scale development and are good representations of what educational researchers would work with. The covariance structure from the multidimensional scale were used to simulate samples for these conditions, and the factor loadings for the unidimensional model were used to simulate samples for this condition. In each case, the goal was to create a short form that is half the length of the long form.

Additional manipulated simulation conditions were the relationship with an external variable and full-scale model misspecification. For the relationship with an external variable, the two levels that were manipulated are (a) no relationship and (b) a moderate relationship (approximately equal to a path coefficient of 0.6 standard

deviations). The full-scale model misspecification was manipulated in the simulation according to three levels: (a) no misspecification, (b) a minor misspecification in the factor loadings (i.e., population models modified to have six of the items cross-loading on a nuisance factor with a loading of 0.3), and (c) a major misspecification in the factor loadings (i.e., same as (b), but with factor loadings of 0.6 on a nuisance factor).

The data were simulated in R v3.5.0 [21] using the ‘MASS’ package [22]). The baseline condition (i.e., high reliability, no external variable relationship, no misspecification) used the values provided by the original models as the population values. These values were changed as necessary to create new population models that fit the target simulated conditions, resulting in fifteen covariance matrices. The sample size of each condition was set to 500. For each combination of manipulated conditions, we created 100 datasets.

3.3 Outcomes

The outcome variables of the simulation were the time to converge to a short form, the average level of model fit of the short form, and the composite reliability of the short form for each factor.

The comparative fit index (CFI), Tucker-Lewis Index (TLI), and Root Mean Square Error of Approximation (RMSEA) were used as the model fit indices, and the cutoff values of CFI > .95, TLI > .95, and RMSEA < .05 were used as indicators of adequate model fit [26].

For this study, the composite reliability of the one and three factor models was calculated as [23]:

$$CR_{\omega} = \frac{(\sum_{i=1}^I \lambda_i)^2}{(\sum_{i=1}^I \lambda_i)^2 + \sum_{i=1}^I \theta_i}$$

where the items are indexed with i , λ are the standardized factor loadings, and θ are the (standardized) residual variances of the items [24]. For the bifactor model, the composite reliability for the of general factor was calculated as

$$CR_{\omega_h} = \frac{(\sum_{i=1}^I \lambda_{gi})^2}{(\sum_{i=1}^I \lambda_{gi})^2 + \sum_{i=1}^I ((\sum_{s=1}^S (\lambda_{si})^2) + \sum_{i=1}^I \theta_i)}$$

where s indexes the specific factors [25].

4. RESULTS

For each factor model, results for the “Minor Error with External Variable” condition did not have noticeable differences from either the “Minor Error with No External Variable” conditions or the “No Error” conditions, so this condition was dropped from the current study.

4.1 One Factor Model

The time to complete for each algorithm was similar across conditions, except for GA (see Table 1), which was faster. The time to converge was slightly longer for the ACO, SA, and TS under the major error with an external relationship condition.

The average model fit statistics for both factor models across 100 replications of the analysis for the three conditions is also shown in Table 1, where bolded values indicate good model fit. When there is no error in the original model, each algorithm produced good model fit, but as the error level increases the model fit decreased.

In the major error conditions, only the SA algorithm had model fit greater than the traditional cutoff values.

Table 1. Model fit of short forms for one factor model

Error/ External	Method	Minutes to Complete	CFI	TLI	RMSEA
None/ No	ACO	2.482	0.976	0.969	0.043
	SA	2.516	0.993	0.992	0.018
	TS	3.933	0.985	0.981	0.028
	GA	0.699	0.975	0.967	0.042
Minor/ No	ACO	2.575	0.961	0.950	0.055
	SA	2.572	0.987	0.983	0.027
	TS	3.987	0.977	0.971	0.036
	GA	0.708	0.964	0.953	0.051
Major/ No	ACO	2.713	0.943	0.926	0.061
	SA	2.581	0.983	0.978	0.029
	TS	3.956	0.940	0.923	0.061
	GA	0.701	0.850	0.807	0.112
None/ Yes	ACO	3.059	0.943	0.929	0.058
	SA	2.871	0.981	0.976	0.029
	TS	2.819	0.923	0.903	0.066
	GA	0.702	0.859	0.823	0.106
Major/ Yes	ACO	3.301	0.942	0.928	0.058
	SA	3.107	0.981	0.976	0.029
	TS	5.162	0.934	0.917	0.060
	GA	0.752	0.855	0.819	0.108

The inclusion of the external variable reduced the model fit for each of the algorithms such that the conditions with no error and an external variable relationship had similar fit to the conditions with major error and either with or without external variable relationship (see Table 1).

Table 2 shows the reliability of the full form of the scale, as well as the reliability the short forms. As expected, the full form resulted in scores with higher reliability than all the short forms. The ACO had the greatest composite reliability for both the no error and minor error conditions, followed by the GA.

Table 2. Composite reliability estimates with one-factor model

Method	Reliability
Full Form	0.889
ACO	0.854
SA	0.813
TS	0.806
GA	0.835

4.2 Three Factor Model

For all five conditions, the SA and TS algorithm took about twice as long to converge on average as compared to the ACO and GA algorithms (see Table 3). For the three-factor model, the average model fit for the conditions with no external variable can be seen in Table 3. In both the no error and minor error conditions, each of the algorithms had good model fit. In the major error conditions, only the SA algorithm produced short forms with adequate model fit according to all three fit indices. The ACO and TS algorithms

had good model fit according to CFI and TLI (ACO) and CFI (TS), while the GA algorithm had poor fit according to all three fit indices.

Table 3. Model fit for short forms with three-factor structure

Error/ External	Method	Minutes to complete	CFI	TLI	RMSEA
None/ No	ACO	2.967	0.979	0.973	0.043
	SA	6.228	0.991	0.989	0.024
	TS	6.063	0.986	0.982	0.031
	GA	2.705	0.977	0.970	0.044
Minor/ No	ACO	3.091	0.978	0.971	0.047
	SA	6.238	0.989	0.985	0.030
	TS	6.264	0.984	0.979	0.037
	GA	2.710	0.974	0.967	0.048
Major/ No	ACO	3.285	0.964	0.953	0.054
	SA	6.295	0.990	0.987	0.027
	TS	6.061	0.956	0.943	0.056
	GA	2.695	0.910	0.883	0.087
None/ Yes	ACO	3.713	0.978	0.971	0.047
	SA	7.040	0.989	0.985	0.030
	TS	7.577	0.984	0.979	0.037
	GA	2.685	0.974	0.967	0.048
Major/ Yes	ACO	3.387	0.960	0.948	0.060
	SA	6.173	0.984	0.979	0.036
	TS	6.586	0.950	0.935	0.063
	GA	2.282	0.917	0.892	0.087

Including an external variable had little effect on the model fit indices (see Table 3). With no error, the average model fit was approximately the same between the no external variable conditions and moderate external variable conditions, while the average model fit somewhat decreased in the major error condition for the external variable conditions as compared to the no external variable conditions. Only the SA produce short forms with good model fit across all the conditions.

The reliability of the full form and short forms with the three-factor CFA is shown in Table 4. All methods produced short forms with less reliable scores than the full form, but among the metaheuristic methods, the ACO produced short forms with the largest composite reliability for each of the factors in each condition.

Table 4. Composite reliability estimates with three-factor model

Method	Reliability Factor 1	Reliability Factor 2	Reliability Factor 3
Full form	0.870	0.910	0.900
ACO	0.788	0.846	0.833
SA	0.752	0.829	0.813
TS	0.754	0.828	0.809
GA	0.763	0.846	0.819

4.3 Bifactor Model

With the bifactor model, the GA had the fastest time to converge, and the ACO took about four times longer. The TS and SA algorithms had convergence times that were about 10 times of the GA algorithm.

Table 5 shows the average model fit indices of the bifactor model for the conditions with no external variable relationship. The SA and TS algorithms produced short forms with good model fit by each fit index in every condition, while the ACO resulted in good model fit by each fit index except for the RMSEA in the major error condition. The GA had good model fit by CFI in the no and minor error conditions only.

Including the external variable tended to reduce model fit. Both the SA and TS showed slight reductions in model fit across both error conditions, but still found short forms with good model fit according to all three fit indices. The ACO maintained approximately the same model fit in both no error and minor error conditions, but showed an increase in average fit in the major error conditions when comparing the no external variable to moderate external variable relationship conditions. However, only the CFI and TLI showed good model fit in these conditions (see Table 5).

The reliability of general factor with the full form and short forms with the bi-factor model are shown in Table 6. For the example scale used in this study, the full form produced scores with adequate reliability for the general factor, but for the specific factors the composite reliability is low. For the short forms, none of the algorithms produced consistently greater reliabilities for every factor in these conditions. The reliability of general factor with the short forms were smaller than the reliability of the full form for all algorithms. The GA performed best for the general factor reliability than the ACO, SA and TS.

Table 5. Model fit of short forms with bi-factor model and external variable

Error/ External	Method	Minutes to complete	CFI	TLI	RMSEA
None/ No	ACO	13.054	0.986	0.976	0.041
	SA	37.925	0.995	0.992	0.020
	TS	39.933	0.992	0.986	0.028
	GA	3.577	0.951	0.939	0.074
Minor/ No	ACO	12.296	0.982	0.969	0.047
	SA	40.658	0.994	0.990	0.023
	TS	48.200	0.990	0.984	0.031
	GA	3.562	0.954	0.943	0.069
Major/ No	ACO	12.875	0.974	0.956	0.055
	SA	39.556	0.993	0.987	0.027
	TS	41.211	0.982	0.969	0.043
	GA	3.606	0.945	0.921	0.079
None/ Yes	ACO	18.537	0.986	0.976	0.041
	SA	45.815	0.995	0.992	0.020
	TS	49.665	0.992	0.986	0.028
	GA	3.458	0.951	0.939	0.074
Major/ Yes	ACO	17.989	0.975	0.959	0.051
	SA	42.336	0.991	0.986	0.028
	TS	38.947	0.979	0.965	0.046
	GA	3.546	0.940	0.907	0.080

For the specific factors, the TS performed better than the other methods for two out of three factors. Surprisingly, the TS produced scores with higher reliability than the full form for factor 3, and the SA produced higher reliability than the full form for factor 2.

The results with the bi-factor model are limited in that the scale used produced scores with low reliability. Using a different scale that results in higher reliability of scores of the full form for all factors might have produced different results with respect to the comparison of algorithms.

Table 6. Reliability of short forms of general factor of bi-factor model

Method	General Factor	Factor 1	Factor 2	Factor 3
Full	0.715	0.304	0.114	0.319
ACO	0.661	0.067	0.115	0.062
SA	0.641	0.061	0.133	0.055
TS	0.659	0.149	0.124	0.433
GA	0.669	0.055	0.118	0.049

5. CONCLUSION

In general, the algorithms produced short forms with adequate model fit in all cases with no error, with two exceptions: each algorithm except the SA under the one factor model with an external variable, and the GA under the bifactor model. Therefore, when the original scale is correctly specified, the results showed that the algorithms are likely to produce short forms with model fit that maintain the desired factor structure of the scale.

The ACO, TS, and GA each had problems maintaining good model fit for the short forms with increasing error, though this was alleviated by increasing the factor structure's complexity. Including an external variable into the process generally had a small negative effect on average model fit, but the effect was never enough to cross the model fit thresholds. Overall, the SA provided short forms with the best average model fit in every single condition, while the ACO seemed to have better reliability on average for each of the factors with one factor and three-factor models. Given that the difference in reliability between the ACO and SA algorithms was about .05 or less on average, the practical difference in reliability between these methods may be outweighed by the difference model fit of the resulting short forms. Therefore, the current results lead to recommending the SA as the preferable metaheuristic algorithm for automated short form selection.

This study provides useful information to applied researchers about the benefits and drawbacks of utilizing these four algorithms for scale short form development in some common scenarios in educational research. This will allow for easier creation of psychometrically-sound short forms with stronger evidence of validity, especially as compared to creating short forms manually. Future research could apply these algorithms to the short form creation problem alongside other methods on real data to compare the efficacy of each approach.

6. REFERENCES

- [1] Fischer, D.G. and C. Fick, *Measuring Social Desirability: Short Forms of the Marlowe-Crowne Social Desirability Scale*. Educational and Psychological Measurement, 1993. **53**(2): p. 417-424.
- [2] Leite, W.L., I.C. Huang, and G.A. Marcoulides, *Item Selection for the Development of Short Forms of Scales Using an Ant Colony Optimization Algorithm*. Multivariate Behavioral Research, 2008. **43**(3): p. 411-431.
- [3] Noble, W., et al., *A short form of the Speech, Spatial and Qualities of Hearing scale suitable for clinical use: The SSQ12*. International journal of audiology, 2013. **52**(6): p. 409-412.
- [4] Dennis, C.L., *The breastfeeding self-efficacy scale: Psychometric assessment of the short form*. Journal of Obstetric, Gynecologic & Neonatal Nursing, 2003. **32**(6): p. 734-744.
- [5] Wester, S.R., et al., *Development and evaluation of the Gender Role Conflict Scale Short Form (GRCS-SF)*. Psychology of Men & Masculinity, 2012. **13**(2): p. 199.
- [6] Lim, S.Y. and E. Chapman, *Development of a short form of the attitudes toward mathematics inventory*. Educational Studies in Mathematics, 2013. **82**(1): p. 145-164.
- [7] Petrillo, G., et al., *The Mental Health Continuum-Short Form (MHC-SF) as a measure of well-being in the Italian context*. Social Indicators Research, 2015. **121**(1): p. 291-312.
- [8] Dréo, J., et al., *Metaheuristics for hard optimization*. 2006, New York: Springer.
- [9] Olaru, G., M. Witthöft, and O. Wilhelm, *Methods matter: Testing competing models for designing short-scale Big-Five assessments*. Journal of Research in Personality, 2015. **59**: p. 56-68.
- [10] Marcoulides, G.A. and Z. Drezner, *Tabu search variable selection with resource constraints*. Communications in Statistics: Simulation & Computation, 2004. **33**(2): p. 355-362.
- [11] Browne, M., M. Rockloff, and V. Rawat, *An SEM Algorithm for Scale Reduction Incorporating Evaluation of Multiple Psychometric Criteria*. Sociological Methods & Research, 2018. **47**(4): p. 812-836.
- [12] Kirkpatrick, S., C. Gelatt, and M. Vecchi, *Optimization by simulated annealing*. Science, 1983. **220**(4598): p. 671-680.
- [13] Colorni, A., M. Dorigo, and V. Maniezzo, *Distributed optimization by ant colonies*, in *Proceedings of ECAL 91 - First European Conference on Artificial Life*, F. Varela and P. Bourguine, Editors. 1992, Elsevier Publishing: Paris, France.
- [14] Yarkoni, T., *The abbreviation of personality, or how to measure 200 personality scales with 200 items*. Journal of Research in Personality, 2010. **44**: p. 180-198.
- [15] Marcoulides, K.M. and C.F. Falk, *Model Specification Searches in Structural Equation Modeling with R*. Structural Equation Modeling: A Multidisciplinary Journal, 2018. **25**(3): p. 484-491.
- [16] Drezner, Z., G.A. Marcoulides, and S. Salhi, *Tabu search model selection in multiple regression analysis*. Communications in Statistics - Simulation and Computation, 1999. **28**(2): p. 349-367.
- [17] Leite, W.L. and S.N. Beretvas, *Validation of Scores on the Marlowe-Crowne Social Desirability Scale and the Balanced Inventory of Desirable Responding*. Educational and Psychological Measurement, 2005. **65**(1): p. 140-154.
- [18] Tschannen-Moran, M. and A.W. Hoy, *Teacher efficacy: Capturing an elusive construct*. Teaching and teacher education, 2001. **17**(7): p. 783-805.
- [19] Reynolds, C.R., R.W. Kamphaus, and K.J. Vannest, *Behavior assessment system for children (BASC)*, in *Encyclopedia of clinical neuropsychology*. 2011, Springer: New York. p. 366-371.
- [20] Splett, J.W., et al., *Factor analytic replication and model comparison of the BASC-2 Behavioral and Emotional Screening System*. Psychological assessment, 2017. **29**(12): p. 1543.
- [21] R Development Core Team, *R: A language and environment for statistical computing*. 2018, R Foundation for Statistical Computing: Vienna, Austria.
- [22] Venables, W.N. and B.D. Ripley, *Modern applied statistics with S*. 2002, New York:: Springer Science & Business Media,.
- [23] Raykov, T., *Estimation of Composite Reliability for Congeneric Measures*. Applied Psychological Measurement, 1997. **21**(2): p. 173-184.
- [24] McDonald, R.P., *Test theory: A unified treatment*. 1999, Mahwah, NJ: Lawrence Erlbaum.
- [25] Gignac, G.E. and M.W. Watkins, *Bifactor modeling and the estimation of model-based reliability in the WAIS-IV*. Multivariate Behavioral Research, 2013. **48**(5): p. 639-662.
- [26] Hu, L. and P.M. Bentler, *Cutoff criteria for fit indexes in covariance structure analysis: conventional criteria versus new alternatives*. Structural Equation Modeling, 1999. **6**: p. 1-55.

Detecting Creativity in an Open Ended Geometry Environment

Roi Shillo
Ben-Gurion University
roishillo@gmail.com

Nicholas Hoernle
University of Edinburgh
nicholas.hoernle@gmail.com

Kobi Gal
Ben-Gurion University
University of Edinburgh
kobig@bgu.ac.il

ABSTRACT

Creativity is a dynamic process which generates ideas that are both novel and of value. However there is little understanding in what drives creativity in students and how to help teachers or education experts to detect creative thinking. This paper begins to address this gap by providing a platform and experiments for studying how creative outcomes can ensue over time. The platform is an open ended environment for creating geometrical shapes that supports exploration and trial and error. We show that participants exhibit diversity in their usage patterns in the system, and in particular, some exhibiting 'creative leaps' in which they move from creating a sequence of shapes in one category to another, new category. We designed a visualization tool that aids understanding in detecting these aspects in students' work. We provide a basic computational model that is able to predict whether a student will create a new shape at a given point in time. The impact of this work is in beginning to provide tools for promoting creativity in students and directing their interactions in a way that facilitates the creative process.

Keywords

Creativity, Divergent Thinking, Geometry, Geogebra

1. INTRODUCTION

There is multiple evidence that exhibiting creativity in the classroom is linked to positive learning gains, and using educational technology to bring about creativity is an active area of research [19, 8]. To date, however, such technologies have relied on human teachers to detect and to promote creative behavior. While research on adaptive technologies for education have flourished, there are few studies on automatically detecting creativity from students' interactions.

This paper begins to address this gap by providing a platform for studying and computationally modeling creative tasks. The task requires participants to create geometric

shapes and explore multiple solutions in a domain that is simple to define and explain, while still providing a rich space of possible solutions.

The task was chosen so that there is no single correct answer, and the goodness of a solution is measured by the number and quality of the different answers. This task supports a process of exploration and discovery. There are many possible strategies for solving the tasks, some requiring more skills than others.

We study people's search trajectories in the space of possible solutions, showing that people exhibit creative leaps [9], alternating between clusters of solutions and exploration. We adapt a model by Leikin [10] for measuring creative outcomes in users' interactions, based on defining their work in terms of flexibility, originality and fluency. We provide a visualization tool that decomposes a user's interaction sequence in the system to separate sequences of solutions. When a creative leap occurs, the solutions in the inferred sequence belong to the same class.

We show that people's creative outcomes in the system varies widely, in a way that depends on the creative leaps that are exhibited by the participants. We built a computational model that attempts to predict whether or not a given shape is new for a given participant. We define several sets of features that include statistics about shapes created as well as GUI operations used to create the shapes. The best performance was achieved by a random forest classifier that was based on both features.

These results can potentially inform the design of algorithms for detecting and promoting creative outcomes.

The remainder of this paper is organized as follows. In the next section we provide a general description of the creativity testbed for creating geometric shapes. We then describe a tool for visualizing the shapes created by individual users over time, and how to cluster these shapes in a way that can detect creative leaps. Finally we provide a computational model for detecting new shapes in a user's interactions, and discuss ways to extend this model to detect creative outcomes.

2. RELATED WORK

Previous research showed that creativity is linked to positive learning gains, and using educational technology to bring

Roi Shillo, Nicholas Hoernle and Kobi Gal "Detecting Creativity in an Open Ended Geometry Environment" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 408 - 413

about creativity is an active area of research [19]. While these programming environment support interactions that can lead to creative thinking, they rely completely on human teachers to detect and to promote this behavior.

We focus on the use of technology to promote divergent thinking, which is a type of creative ability that generates multiple answers to problems [18, 6]. Guilford [6] defines divergent thinking as generation of multiple answers to a problem. Torrance[17] defined fluency, flexibility, novelty, and elaboration as parameters that describe divergent thinking. Fluency is the ability to create large number of ideas for a problem that are useful. Flexibility is the ability to change direction, thinking strategies and point of views. Originality is the ability to generate novel and unconventional ideas. Unconventional ideas defined as statistically infrequent.

There are several works focusing on the use of technology to promote creative outcomes in students. Multi Solution Tasks (MST) is mathematics open ended problems with multiple correct answers that can reached in different ways [12]. MST improves the participant's understanding and the connection between his knowledge domains, skills and strategies [4]. Levav-Waynberg & Leikin [12] found that MSTs raised the connection between knowledge domains in geometry, and improve the fluency and flexibility of the participants. Geometry plays a major role in Math teaching. It includes visual, abstract and logical skills. Geometry created opportunities for investigation, generalization, deduction and gives autonomy to the learner to explore mathematics with his personalized preferences [12, 3].

Sophocleous & Pitta-Pantazi [16] found that using software environment for geometry enhanced the creative abilities of students by facilitating them to provide more, different and unique solutions.

Noy et al. [15] demonstrated the role of creative leaps in two dimensional geometry. They show that human players exhibit two types of exploration: 'scavenging', where similar shapes are accumulated, and 'creative leaps', where players shift to a new region in the shape space after a prolonged search. They show that the network of shapes created by human participants is different from the class of networks created by applying a simple random-walk algorithm. We extend their work in two ways. First by providing a computational model to detect new shapes; second in extending the notion of creative leaps to a framework that allows a richer set of actions to be created. In subsequent work they studied creative exploration using a scale invariant model that considers relative changes in signals [7].

There are few studies on automatically detecting creativity from students' interactions. An exception is Manske and Hoppe [14], who have used supervised learning methods to detect creativity in programming assignments that require mathematical skills. They combined low level features (e.g. code snippets) with higher level features (e.g., the use of recursion, number of lines of code) to train numerical predictors and predicted a creativity score for new solutions. Chuang et al.[2] used fuzzy logic to detect student's creativity measures in a gaming environment. Loveless et al.[13] studied the use of technology to promote aspects of cre-

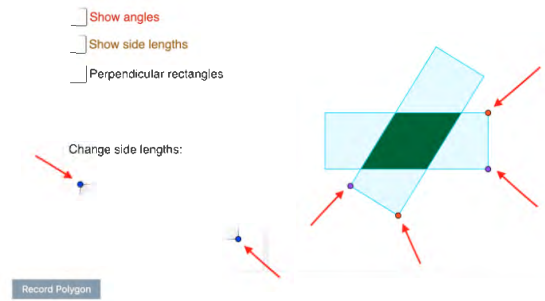


Figure 1: Open Creativity App

ative thinking for student teachers, including the development of new ideas, modifying and evaluating the originality and value of work as it develops. These works relied on manual approaches for detecting creativity and did not study how to visualize these creative outcomes.

3. THE GEO CREATIVITY TESTBED

We designed an activity (built as a GeoGebra app) in which participants create geometric shapes by manipulating the shaded area that intersects two rectangles (See Figure 1).

Participants can employ geometric transformations on each rectangle according to several possible actions supporting by the testbed GUI: Translation (shifting a rectangle along the x or y axis), rotation (re-positioning the rectangle by changing one or more of its angles), re-sizing (increasing or decreasing the size of both of the rectangles). Creating different shapes requires to master different skills. It is easier to create polygons of varying number of sides by rotating the rectangles, but other types of shapes requires more steps, using more actions or have a precise positioning of the rectangles in designated angles.

To help with the positioning, participants can optionally choose to display the angles formed at the vertices, the length of the intersection shape sides, or position the two rectangles perpendicularly one to each other. At any point in time, participants can choose to submit their shape to a gallery. When the shape submitted to the gallery, the interface doesn't change and the participants continue their work from the same point.

The GUI design supports several key factors that have been shown to facilitate creative outcomes in students. First, it provides an open ended task in which there is no single correct answer, and the goodness of a solution is measured by the number and quality of the different answers [10].

Second, the task supports a process of exploration and discovery. Participants can manipulate two rectangles to create new shapes. Trial and error is a key part of the exploration process [5].

Third, understanding the task does not rely on complex or unique tool set. Participants with little knowledge in geometry can use the task and think about novel and useful categories. Participants with more knowledge and experience will have a better potential to think about new and

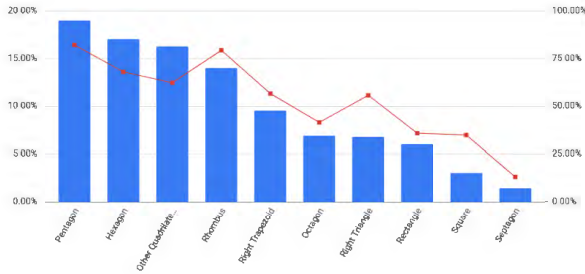


Figure 2: Categories Distribution over Participants

unconventional categories [1].

Fourth, there are many strategies to create shapes from the same category. Some requiring more skills or using more tools than others. For example, squares can be created by rotating and positioning the two rectangles, and also by using “perpendicular rectangles” option in the GUI menu [1].

This task is recommended activity by content developers for K4-K5 students that learn geometry. The idea is for students to learn to generate geometric shapes in a novel way (intersecting the two rectangles). The number of possible shapes that students can create with the system is not bounded.

3.1 Procedure

We recruited 183 Participants (87 undergraduate students and 96 Mturk workers with varying educational background - high school, Bachelor and master graduates). All participants needed to pass a tutorial and comprehension quiz about the study in order to participate.

All participants were requested to “create as many different polygons types as possible by intersecting two rectangles.” Students performed the task as part of an extracurricular activity and were not monetarily compensated. Participants in Mturk received monetary compensation as follows: 30 cents as a show up fee and 10 cents for every type of shape that they’ve created.

The task was limited by 10 minutes, and on average participants spent 4.5 minutes on the task.

Participants created between one and ten different shape categories (e.g., Polygon, square, etc.) and 1445 shapes in total. Figure 2 shows the frequency of the shape categories submitted by users to their portfolio. The x axis denotes the shape category; the blue bars display the number of shapes that was created for each category (values in the left vertical axis). The red line shows the percentage of participants that created shapes of the given category (values in the right vertical axis).

As shown by the figure, the most popular shape categories were Pentagon, Hexagons and “other” Quadrilaterals (quadrilaterals which were not rhombus, square or rectangles). The least popular shape categories were septagon, square and rectangles. There was a general correspondence between the number of times a shape was created and the number of users who created the shape. However, some shapes cat-

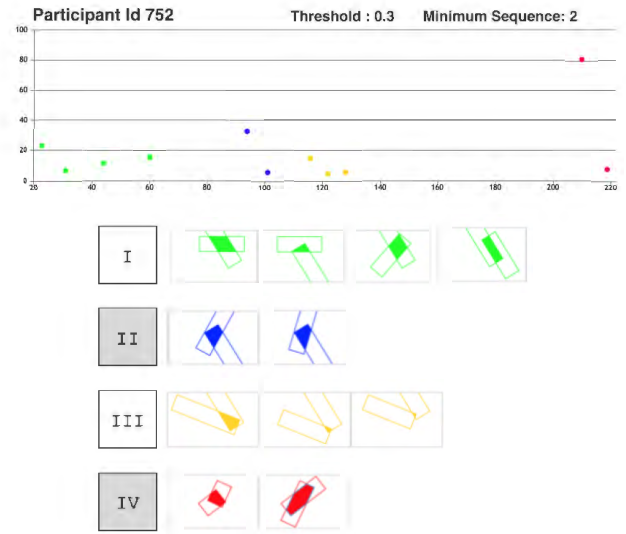


Figure 3: Visualization tool showing timeline of shapes (top), and shape sequences (bottom) for user ID 752

egories, namely Rhombus, square and rectangle, were more popular. For example, the Rhombus is the fourth most popular shape category, yet it is the second most popular shape among the users. We will show later that these shapes played a special part in people’s creative process.

3.2 Visualization Tool

We designed a visualization tool for studying how individual participants create shapes over time. Fig 3 shows the main interactive panel in the visualization system. The main panel shows the shapes created by an individual user (ID 752). The x axis represents time (in seconds) from commencement of the interaction, while the y axis represents the length of time (in seconds) it took to create shapes. For example, the coordinate (30, 4) shows the first shape created by the user (Rhombus) at time 30 and took 4 seconds to create. As shown in the figure, participant ID 752 created 11 shapes over a time span of 230 seconds. We can see that the participant exhibited high variance in the creation time of the shapes. For ease of analysis, there is a way to group shapes into temporal sequences according to the following criteria: First, a sequence has to include at least n contiguous shapes. Second, the probability that the next action belongs to the same sequence is greater than a designated threshold T . The threshold can be set in a way that maximizes the number of shapes in the sequence. A shape i commences a new sequence if $P(\frac{\Delta_i - \mu}{\sigma}) > T$ where Δ_i is the length of time spent to create shape i , μ and σ are the average time and standard deviation for creating shapes by the participant. In this way we consider the extent to which the creation time of each shape agrees with the individual participant (assuming a normal distribution over creation time).

Figure 3 shows four shape sequences that were created by the user, inferred by the criteria described above. We can see that sequence II and IV are relatively short and include shapes that share a common category (that of trapezoids or

hexagons, respectively). In contrast, sequence I and III are longer and include shapes of different categories. We next show how these sequences yield insights into creative aspects of the shape creation processes.

3.3 Measuring Creativity

We measure the creativity of a participant following Torrance [17] who defined dimensions of flexibility, fluency and originality to describe creative solutions to problems: Fluency is the number of solutions that was proposed by the participant. Flexibility is the number of strategies that the participant applies for the solution. Originality is statistically infrequent ideas that were produced by students relative to their classes or groups.

We calculate the score by the formula suggested by Leikin [11], adapted to the geometry app. The flexibility of a shape i is 10 if the shape is new for this participant, or 1 otherwise. The originality of a shape i is 10 if the shape was chosen by fewer than 15% of participants (septagons), 1 if the shape was chosen by between 15% and 40% of participants (rectangles and squares), and 0.1 if the shape was chosen by more than 40% of participants.

The fluency of a shape is always 1. Let n be the number of shapes created by the participant (also the participant's fluency score). The creativity score of a participant is computed as $\sum_{i=1}^n FX_i \cdot OR_i$ where FX_i and OR_i are the flexibility and originality of shape i .

4. CREATIVE LEAPS

Participants exhibited a diverse range of creativity scores in their work. The average score was 80 with a standard deviation of 54. To explain differences between students, we need to analyze the dynamics of how shapes were created. We will distinguish between two types of shape creation, those representing exploration in the space of possible concepts, and those representing exploitation of one of the concepts that is used to create shapes.

Koestler [9] describes a creative leap as the moment where a new dimension of possibilities appears. The creative leap signifies a point in the search space in which the learner discovers a new class of solutions and begins to exploits this space by creating shapes.

In sequence II and IV of participant 752 (shown in Figure 3) the participant exploits the concept of trapezoid and of hexagons, completing two shapes in each category. Both of these sequences represent creative leaps. In contrast, sequence I and III for this user represents exploratory behavior in which the participant does not converge on a shape category. In particular, sequence IV commenced 81 seconds after the last shape, suggesting a lengthy process of exploration leading to the next shape category.

We use the concept of creative leaps to distinguish between different types of participants, as determined by their interaction in the system.

Another example, participate 651 holds an MA degree, exhibited a creativity score of 128, with fluency of 36, flexibility of 8, and originality of 1 (rectangle and square). Participant

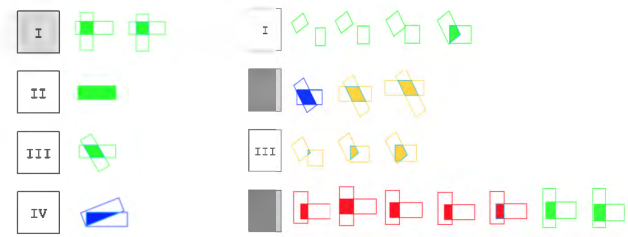


Figure 4: Shape sequences for low scoring participant ID 655 (left) and high scoring participant ID 651 (right)

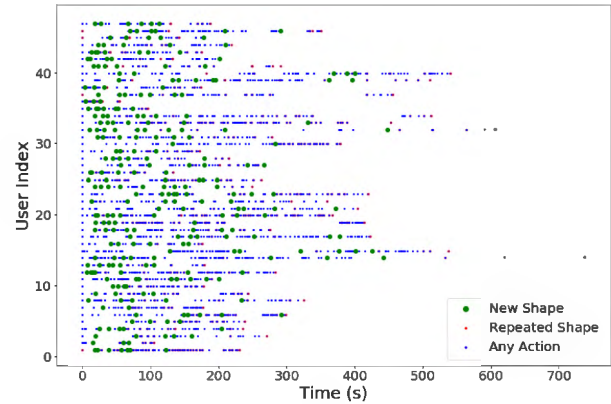


Figure 5: New and old shapes

655 holds a high school diploma, with a creativity score of 61, with fluency of 5, flexibility of 4 and originality of 1 (rectangle and square).

Figure 4 shows the shape sequences for these two participants. As shown by the figure the users exhibited drastically different interaction styles with the system. The low scoring participant (user ID 655) created only five shapes, which can be described by 4 sequences. Only sequence I for this user consisted of shapes of a similar category. In contrast, the high scoring participant (user ID 651) created 36 shapes, which can be described by 10 sequences (for brevity we only show the first four). Five of these sequences included shapes with particular categories.

5. COMPUTATIONAL MODEL

Using the creation of a new shape as a proxy for creativity, we build a computational model that attempts to predict whether or not a given shape is new for this participant. This is a first step for detecting creativity for this participant. To this end, we extract a number of features and use logistic regression and a random forest classifier to predict the binary outcome of whether or not a given shape is new for the user.

Figure 5 shows the layout of new shapes (green) and old shapes (blue) for users across time. It demonstrates that while the proportion of new shapes being generated certainly does decrease as the user continues to interact on the platform, there are a significant number of shapes that are created 'later' on in the user's interaction session. These shapes

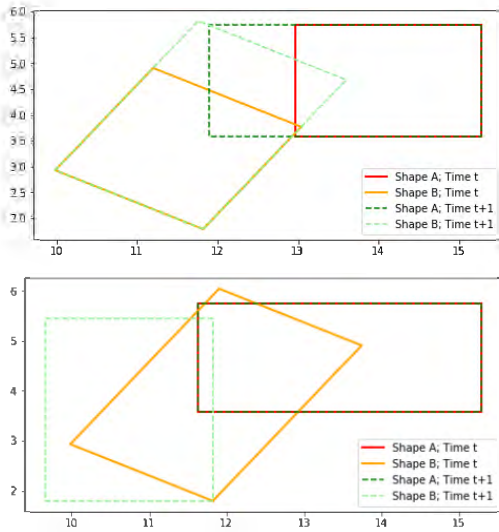


Figure 6: Plots showing the raw shape coordinates and the extracted actions from the database. Top is a resize action where both input rectangles are affected. Bottom is a rotation action to the second rectangle

are new to the user and can signify the commencement of a creative leap.

5.1 Feature Extraction

The raw data represents individual coordinates of the two input rectangles where the user can rotate, translate and resize the two rectangles. We extract changes to these coordinates to represent the 5 actions that the user can perform on the system. Figure 6 shows the primary actions that a user can perform. The resize action affects both shapes simultaneously whereas the user can perform rotate and translate actions to each shape individually.

As a number of actions are preformed before a user submits a shape, we include the history of actions in the feature matrix considering a maximum of 15 actions before the shape was submitted; 95% of shapes were submitted with less than 15 steps. We aim to infer whether including the history of actions provides additional information to a model predicting the creation of a new shape. In other words, our goal is to determine if combinations of actions are predictive of a new shape and thus indicative of certain creative insights.

5.2 Method

Given four feature sets, we use cross-validated logistic regression (LR) and random forest (RF) classification to predict the binary classification problem of whether or not the shape is new for the user. Other classification techniques were explored including support vector machines, boosted decision trees and naive Bayes classifiers but LR and RF were chosen for simplicity and the ease of interpretation into the parameter coefficients (or feature importance in the RF). Moreover, LR allows the application of an L1 sparsity regularizing parameter to induce sparsity in the feature space (thereby assisting inference). The available features are:

Table 1: Different combinations of available features that the four feature sets used.

Feature Set	Shape History	Action History	Aggregated Action History
Feature set 1	Y		
Feature set 2	Y	Y	
Feature set 3	Y		Y
Feature set 4	Y	Y	Y

1. **Shape History.** Counts of previously submitted shapes from the user for each of the 10 shape categories.
2. **Action History.** A one-hot encoded representation of actions in 15 previous steps leading into the shape creation. Following the actions shown in Figure 6 there were 3 possible actions (translate, rotate or resize) at each step. We further include a 'control' action that represents a recorded step but no action. This case might occur when the user does not interact with the shapes but rather interacts with a different UI element or possibly performs an administrative action such as submitting the shape. This creates 60 features in total for each instance.
3. **Aggregated Action History.** The sum of the **Action History** features across the 15 steps. This results in 4 additional features per instance describing the number of times each action was performed. For example, a vector (2, 3, 5, 5) corresponds to 2 translate, 3 rotate, 5 resize and 5 control actions (in any order).

The three sets of available features were aggregated in different combinations to provide four feature sets that were used for evaluation. Table 1 summarizes the different combinations that were used.

5.3 Results

Our goal was to determine if certain sequences and/or combinations of actions are predictive of the user generating a new shape. Table 2 summarizes these results. We note that the Feature Set 3 provides a slight improvement over the baseline of Feature Set 1 for both LR and RF. Feature Set 4 with RF shows the greatest accuracy with a 2% increase over the baseline feature set. The results are reported from a 10 fold cross-validation but the predictive increase is not significant across the folds. The results suggest that the inclusion of the action data does assist slightly with the predictive performance of the model but we note the result is not conclusive. However, analyzing the feature weights of the LR for Feature Set 3 and the RF for Feature Set 4 is illuminating.

It is interesting to note that the RF model with the temporal features outperforms the LR on these same feature sets. Again, although the results are not significant they do suggest there is a more complex interaction of the action history of the user that might be predictive of the new shape creation. Further investigation into how **sequences** of actions might be indicative of the creation of a new shape (and thereby indicative of a creative leap) is needed to answer this question definitively.

Table 2: Table showing the results from the prediction task of predicting a new shape for a given user.

Feature Set	Accuracy	
	LR	RF
Trivial all 0 prediction	59.9%	59.9%
Feature set 1	75.5%	75.4%
Feature set 2	74.8%	76.1%
Feature set 3	76.0%	76.5%
Feature set 4	73.5%	77.5%

6. CONCLUSION AND FUTURE WORK

We presented an approach for studying creativity using a web based tool in which participants created geometric shapes. This task supports a process of exploration and discovery, allowing people to exhibit creative leaps in which they transition between different areas of the search space of possible solutions. We adapted a model by Leikin for measuring creative outcomes in users' interactions. We collected data from multiple people interacting with the system showing that they vary widely in terms of the creativity they exhibit. We built a visualization tool that decomposes a user's interaction sequence in the system to separate sequences of solutions. We built a computational model that attempts to predict whether or not a given shape is new for a given participant. We define several sets of features that include the number and categories of shapes that were created by the user, as well as basic actions performed by the users in the system for a window of activity. The best performance was achieved by a random forest classifier that was based on both features. In future work we intend to extend the computational model to detecting creative outcomes in new types of domains.

7. ACKNOWLEDGMENTS

The authors would like to thank Yuval Hart for very useful discussions and advice; Sara Hershkovitz, Guy Hed and Shoshana Gilead from the Center of Education Technology, for developing the task, their pedagogic support and helping with the coding; Adar Slonim and Ahmad Majadly for development help.

8. REFERENCES

- [1] S. W. Chae, Y. W. Seo, and K. C. Lee. Task difficulty and team diversity on team creativity: Multi-agent simulation approach. *Computers in Human Behavior*, 42:83–92, 2015.
- [2] T.-Y. Chuang, E. Z.-F. Liu, and W.-Y. Shiu. Game-based creativity assessment system: the application of fuzzy theory. *Multimedia Tools and Applications*, 74(21):9141–9155, 2015.
- [3] D. H. Clements and M. T. Battista. Geometry and spatial reasoning. 1992.
- [4] J. Dhombres. Is one proof enough? travels with a mathematician of the baroque period. *Educational Studies in Mathematics*, 24(4):401–419, 1993.
- [5] C. Granberg and J. Olsson. Ict-supported problem solving and collaborative creative reasoning: Exploring linear functions using dynamic mathematics software. *The Journal of Mathematical Behavior*, 37:48–62, 2015.
- [6] J. P. Guilford. The nature of human intelligence. 1967.
- [7] Y. Hart, H. Goldberg, E. Striem-Amit, A. E. Mayo, L. Noy, and U. Alon. Creative exploration as a scale-invariant search on a meaning landscape. *Nature communications*, 9(1):5411, 2018.
- [8] S. Hershkovitz, I. Peled, and G. Littler. Mathematical creativity and giftedness in elementary school: Task and teacher promoting creativity for all. *Creativity in mathematics and the education of gifted students*, pages 255–269, 2009.
- [9] A. Koestler. The act of creation. 1964.
- [10] R. Leikin. Exploring mathematical creativity using multiple solution tasks. *Creativity in mathematics and the education of gifted students*, 9:129–145, 2009.
- [11] R. Leikin. Evaluating mathematical creativity: The interplay between multiplicity and insight. *Psychological Test and Assessment Modeling*, 55(4):385, 2013.
- [12] A. Levav-Waynberg and R. Leikin. The role of multiple solution tasks in developing knowledge and creativity in geometry. *The Journal of Mathematical Behavior*, 31(1):73–90, 2012.
- [13] A. Loveless, J. Burton, and K. Turvey. Developing conceptual frameworks for creativity, ict and teacher education. *Thinking Skills and Creativity*, 1(1):3–13, 2006.
- [14] S. Manske and H. U. Hoppe. Automated indicators to assess the creativity of solutions to programming exercises. In *2014 IEEE 14th International Conference on Advanced Learning Technologies*, pages 497–501. IEEE, 2014.
- [15] L. Noy, Y. Hart, N. Andrew, O. Ramote, A. E. Mayo, and U. Alon. A quantitative study of creative leaps. In *ICCC*, pages 72–76, 2012.
- [16] P. Sophocleous and D. Pitta-Pantazi. Creativity in three-dimensional geometry: How an interactive 3d-geometry software environment enhance it. In *Proceedings of seventh conference of the European Research in Mathematics Education*, pages 1143–1153, 2011.
- [17] E. P. Torrance. *Torrance tests of creative thinking: Norms-technical manual: Verbal tests, forms a and b: Figural tests, forms a and b*. Personal Press, Incorporated, 1966.
- [18] D. J. Treffinger, G. C. Young, E. C. Selby, and C. Shepardson. Assessing creativity: A guide for educators. *National Research Center on the Gifted and Talented*, 2002.
- [19] S. Wheeler, S. Waite, and C. Bromfield. Promoting creative thinking through the use of ict. *Journal of Computer Assisted Learning*, 18(3):367–378, 2002.

Tutorbot Corpus: Evidence of Human-Agent Verbal Alignment in Second Language Learner Dialogues

Arabella Sinclair
University of Edinburgh
10 Crichton Street
Edinburgh, Scotland
s0934062@sms.ed.ac.uk

Kate McCurdy
University of Edinburgh
10 Crichton Street
Edinburgh, Scotland
s1841537@sms.ed.ac.uk

Christopher G. Lucas
University of Edinburgh
10 Crichton Street
Edinburgh, Scotland
clucas2@inf.ed.ac.uk

Adam Lopez
University of Edinburgh
10 Crichton Street
Edinburgh, Scotland
alopez@inf.ed.ac.uk

Dragan Gašević
Monash University
Melbourne
Australia
dragan.gasevic@monash.edu

ABSTRACT

Prior research has shown that, under certain conditions, Human-Agent (H-A) alignment exists to a stronger degree than that found in Human-Human (H-H) communication. In an H-H Second Language (L2) setting, evidence of alignment has been linked to learning and teaching strategy. We present a novel analysis of H-A and H-H L2 learner dialogues using automated metrics of alignment. Our contributions are twofold: firstly we replicated the reported H-A alignment within an educational context, finding L2 students align to an automated tutor. Secondly, we performed an exploratory comparison of the alignment present in comparable H-A and H-H L2 learner corpora using Bayesian Gaussian Mixture Models (GMMs), finding preliminary evidence that students in H-A L2 dialogues showed greater variability in engagement.

Keywords

Language learning, chatbot, dialogue, alignment, tutoring, agent, second language, student engagement, assessment

1. INTRODUCTION

This work reports on evidence of alignment within student dialogue to that of an automatic tutor even when both parties are restricted in their capacity to align: the student as an L2 learner may lack the linguistic proficiency to show alignment [5], and the agent aligns only minimally by design. Alignment consists of interlocutor interaction adaptation, resulting in convergence, or in their sharing of the same concept space [13, 8]. Alignment of student to tutor in dialogue has been used as a predictor of both student learning and engagement [20]. A key aspect of dialogue is

the speakers' ability to align: to either show engaged, willing behaviour, or display little discernible adaption to their interlocutor. Interestingly, humans have been shown to exhibit greater alignment to agents than to other humans [4, 6]. In an automated L2 tutoring setting, where students have been shown to imitate tutors as part of their learning process [10] it is of great interest to determine whether the user/learner is actively engaged, simply gaming the system, or disengaged, either because of lack of ability or motivation [1]. Modelling alignment of student to tutor as evidence of engagement could serve as a useful tool in the design of tutor intervention or student assessment since there has been limited research into identifying signs of engagement or gaming in the automated L2 tutoring setting.

Given this relevance of alignment in modelling engagement during tutor-student L2 dialogues [20], one key question is whether L2 students demonstrate alignment behavior in conversation with an automated dialogue agent, even when they know the agent is not human. Prior work has established that L2 students display alignment when conversing with a human tutor, in Human-Human (H-H) interactions [17]; however, this work has also demonstrated relatively *symmetric* alignment, as human tutors verbally aligned with their students in turn — this raises the possibility that L2 learners may fail to display alignment if the dialogue is predominantly *asymmetric*, when interacting with an agent whose capacity to align is also limited. Studies of Human-Agent (H-A) dialogues in other domains demonstrate that fluent speakers verbally align with agents [4, 6], but given the unique constraints affecting alignment in L2 dialogue [5], we cannot assume that L2 students will behave similarly. If they do, a second key question arises: do L2 students display similar alignment behavior in H-H and H-A dialogues? Even if students align in both contexts, exploratory analysis may reveal critical differences which could inform educational researchers and practitioners working with dialogue agents. Hence, our work addresses the following research questions: **RQ1** *Do L2 students show alignment to an automated dialogue agent (i.e. H-A alignment)?* and **RQ2** *What is the nature of the alignment found in the H-A corpus and how does it differ from that of H-H dialogues?*

Arabella Sinclair, Kate McCurdy, Adam Lopez, Christopher G. Lucas and Dragan Gasevic "Tutorbot Corpus: Evidence of Human-Agent Verbal Alignment in Second Language Learner Dialogues" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 414 - 419

We present a study of student verbal alignment within a new dialogue corpus consisting of transcripts from a language teaching app where students are interacting with a dialogue agent. We contrast this H-A corpus with a comparable H-H L2 learner corpus of tutoring dialogue transcripts. We found that students in H-A interactions align to the agent more so than they would by chance, albeit to a lesser degree than students in H-H dialogues. Our results found that within H-H dialogues, students exhibited greater alignment than tutors. Finally, we compared the distribution of student to tutor alignment within both corpora, revealing more variance in alignment within the H-A dialogues. We hypothesise this was due to either student engagement effects, or different types of student alignment strategy within the H-A dialogues than the more uniform alignment present in the H-H corpus.

2. BACKGROUND

To achieve effective communication within dialogue, speakers typically align, adapting their interaction to their interlocutor. The Interactive Alignment Model (IAM) [13], describes this process as that of speakers agreeing on a shared conceptual space. In educational settings, by contrast, alignment has been found to predict both student learning and engagement [20]. Automatic alignment between interlocutors occurs over different linguistic levels, including that of the lexical, syntactic and semantic [13]. *Lexical* alignment consists of speakers beginning to use the same words [21, 17] or phrases [6] as each other. *Syntactic* alignment consists of the use of the same parts of speech patterns, such as similar noun-phrase constructions, or similar adjuncts [14] as the conversation progresses. Finally, *semantic* alignment can range from adaptation to individual differences in personality [11] to convergence at a higher level of representation such as Dialogue Acts [16]. Recent research has established a number of metrics for linguistic alignment which can be computed automatically, enabling large-scale corpus analysis based on sequential pattern mining [6]. These methods quantify alignment in terms of the *expressions*, or contiguous sequences of tokens appearing in the utterances of both interlocutors. While these methods have been applied to the analysis of H-A interaction [6] and H-H student-tutor interaction [17], the work presented in this paper is the first to apply this computational methodology to compare H-A and H-H dialogue in an educational L2 setting.

Within an L2 practice setting, we predict alignment to have slightly different properties compared to a fluent conversational setting where speakers tend to have a symmetric contribution and equal status within the dialogue [18], and are equally capable of participating [5]. L2 learners have been found to perform at a higher level when speaking in dialogue with a peer than in a monologue context [15]. This suggests students draw from the example language of their interlocutor leading us to expect evidence of alignment. L2 students have also been shown to learn vocabulary through taking part in dialogue [9], suggesting this process of alignment and repetition of their interlocutor's speech produces learning gains. In the case of the tutor, their need to adhere to the ZPD suggests that their alignment patterns will also differ from that of straightforward dialogue. These different factors influence the speakers' convergence to a shared mental state [5]. Vygotsky's theory of ZPD [19] states students

Table 1: Tutorbot dialogue example. Italics indicate Expression Repetition

1.	bot:	What is your <i>favorite day of the week</i> ?
2.	user:	My <i>favorite day of the week</i> is Friday ...
3.	bot:	Do you play sports ?
4.	user:	yes
5.	bot:	What sport do you <i>play</i> ?
6.	user:	I <i>play</i> volleyball and I go running
7.	bot:	When do you do that ?
8.	user:	On Monday , Wednesday and Friday
9.	bot:	What time does it start ?
10.	user:	At 4 o'clock in the afternoon

will learn best when addressed at the correct level, therefore we also expect to see alignment, in the case of tutors in H-H dialogues, to student ability.

3. CORPORA

We are interested in the comparison between student alignment in H-H and H-A dialogues. The H-A corpus analyzed in this study comprises dialogues drawn from a large-scale commercial platform for L2 learners¹. In this application, novice learners of English who had completed lessons on relevant topics were offered the possibility to review the material via simple conversations with the automated dialogue agent Tutorbot. Given the focus on relevant learning material, the agent engaged learners in a system-initiative dialogue with extensive guidance, rather than user-initiative [2]; as a result, Tutorbot steered the learner conversations very deliberately, and alignment from the tutor agent to the student was highly limited by design. A sample dialogue from the corpus can be seen in Table 1. The H-H corpus used is the Barcelona English Language Corpus (BELC) [12] which consists of tutor guided conversations with L2 learners of English at varying stages of fluency from absolute beginner to approaching intermediate. The tutor's goal was to elicit as much conversation from the learner as possible while setting them at ease in as natural and conversational a manner as they could. Key differences are shown in Table 2. However, it should also be noted that the Tutorbot corpus only consists of single utterance turns, whereas BELC has multiple. The topics are also more diverse in BELC, as the Tutorbot explicitly guided learners to review practiced material rather than engage in open-ended discussion. Nevertheless, certain main topics (*how are you, where are you from, tell me about your family, hobbies, what time do you do that*) and the beginner/lower-intermediate range of learner ability are common to both, facilitating automated alignment comparison.

4. METHODS

4.1 Alignment

In order to analyse the verbal alignment present in both corpora, which allows us to answer both *RQ1* and *RQ2*, we use the expressions-based measures introduced by [6]. This approach identifies sequences of tokens (*Expressions*) which are used by both dialogue participants (thus *established* as expressions). These expressions allow us to see the fixed expressions established between speakers, called the routiniza-

¹This data was kindly shared with us by Babbel, <https://www.babbel.com/>

Table 2: H-A and H-H Corpora Differences

	Tutorbot	BELC
number of dialogues	3689	118
average Num. utterances	20.41	130.69
average Num. tokens	128.99	634.28
average tokens/utterance	6.32	4.85
communication medium	typed	spoken
speakers	H-A	H-H
student L1	German	Spanish
vocabulary overlap	0.085	0.251

tion process in the interactive alignment theory [13], and thus an indication of speaker alignment. We re-define the following in order to discuss our results in the following sections:

Expression Lexicon EL is the set of expressions used by both speakers for a given dialogue.

Expression Variety (EV) is the size of the EL normalised by the total number of tokens in the dialogue. This ratio indicates the variety of the expression lexicon relatively to the length of the dialogue: the higher the EV, the more incidence of established expressions between participants. The EV indicates the routinization between speakers.

$$EV = \frac{\text{length}(EL)}{\text{number of tokens}}$$

Expression Repetition – speaker (ER_S) is the ratio of Expressions to dialogue produced. This is measured in tokens. This value indicates the Expression repetition present in the dialogue, i.e. the higher the ER, the more the speakers dedicate tokens to the repetition of established expressions. This is indicative of speaker alignment.

Initiated Expression (IE_S) are the established expressions initiated by S

Vocabulary Overlap (VO) is the ratio of shared tokens between interlocutors S₁ and S₂. The higher the VO, the more vocabulary is shared between speakers.

$$VO = \frac{(\text{Tokens}_{S_1} \cap \text{Tokens}_{S_2})}{(\text{Tokens}_{S_1} \cup \text{Tokens}_{S_2})}$$

4.2 Baseline

In order to test that the alignment reported was not simply due to corpus-specific vocabulary effects (which would be influenced by the vocabulary overlap defined in the previous section), a ‘scrambled baseline’ was created for each corpus. This was achieved by creating a ‘bag of words’ of the tokens produced by each speaker for a specific dialogue, then substituting each token from each speakers utterances with one from the shuffled bag of words. This method retains the turn-taking of the speakers, and the distribution of utterance lengths from the original dialogue, but removes any word ordering present. In the results section for each alignment measure, we report on whether the effects were significantly different from this baseline. This baseline allows us to compare the effects of alignment across corpora, answering *RQ1*.

4.3 Alignment Distribution Clustering

In order to answer *RQ2* investigating student alignment differences within and between the H-H and H-A corpora, we fitted a Gaussian mixture model (GMM)[7] to the student ER_S data for both the H-H and H-A students. GMMs allowed us to detect and characterize distinct sub-populations within a larger group, provided those sub-populations were marked by differences in a parameter of interest, e.g., measured ER_S. To find the number of components which best fitted the data, we used a Bayesian Gaussian mixture model with a Wishart prior of $[[0.1]]$ on the precisions and a scale-1 exponential prior on the number of clusters, and selected the most probable number of clusters given the data (i.e. the posterior mode), assuming that up to seven clusters might be present. We used a Bayesian approach in order to avoid the degeneracies that are common when using maximum-likelihood estimation and information criteria (e.g., AIC or BIC) to estimate cluster counts and parameters [3]. To implement this, we used the toolkit scikit-learn², package *BayesianGaussianMixture*; the priors on component means were scikit-learn 0.20 defaults.

5. RESULTS AND ANALYSIS

The following subsections all contribute to answering *RQ1*, through the comparison of H-H to H-A student alignment and corpus statistics. Section 5.5 specifically explores the variation in alignment styles across corpora, allowing us to answer *RQ2*.

5.1 Expression Lexicon

The Expression lexicon is the set of expressions which are shared between speakers. On inspection, the most common multi-word expressions being aligned to in the Tutorbot corpus fell into two main categories: 1) the student using the direct re-form of the question in the creation of their answer: “*bot|4: What **is** your **favorite day of the week** ? user|5: My [**favorite day of the week**] [**is**] **Friday**”.* 2) The student reflecting the question back to the tutor-bot. “*bot|4: Where **do you live**? user|5: I live in <LOCATION>, where [**do you live**]?”*. The rephrasing in BELC is different: it is more likely that the tutor will re-phrase the student’s single or multi word answer as a form of confirmatory feedback. e.g. “*Tutor: you like going out with your friends, good*” when this is really more repetition/confirmation. The student alignment also consisted of their reflection of tutor questions back to them, and in their repetition of tutor scaffolding moves (something not present in the Tutorbot corpus due to the agent dialogue design) Table 3 contains details of the vocabulary overlap, speaker specific token ratios and the expression lexicon size differences between corpora.

Table 3: Corpora Differences- values represent the average per dialogue

	Tutorbot	BELC
Expression Lexicon Size (ELS)	3.04	48.55
S1/tokens (%)	0.81	0.68
S2/tokens (%)	0.19	0.30
Voc. Overlap	0.085	0.251
Voc. Overlap S1	0.105	0.312
Voc. Overlap S2	0.258	0.613

²<https://scikit-learn.org/stable/>

5.2 Vocabulary Overlap

The vocabulary overlap (VO) between speakers gives us an idea about how likely ‘alignment’ according to our metric will occur by chance. The results in Table 3 therefore can inform our interpretation of the levels of ER_S reported in section 5.4. Student VO in BELC (HH) is much higher than from the students in Tutorbot (HA) (0.613 vs. 0.258). This could be due to the fact that Tutorbot learners were at a lower level of proficiency, so they did not use such extensive vocabulary; alternatively, it could be due to the method of data collection: Tutorbot allows learners a one turn response (a single utterance), limiting their production.

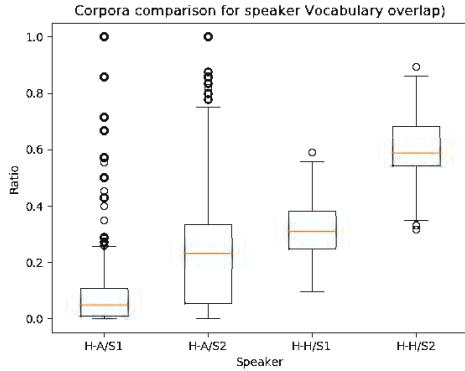


Figure 1: H-H/A corpora Vocabulary Overlap. Speaker difference was significant for H-A ($p < 0.0001$) ($statistic = 6.42, pvalue = num1.4e - 10$) and H-H ($p < 0.001$) ($statistic = -2.11, pvalue = 0.00036$) S1 = Tutor/Agent, S2 = Student

5.3 Expression Variation

We compare the H-H and H-A corpora of real interactions to each other, and to the baseline H- H_R and H- A_R corpora to control for vocabulary effects. Firstly, EV was significantly higher for the H-H corpus ($mean = 0.075, std = 0.025$) than that in the H-A corpus ($mean = 0.032, std = 0.046$). Statistical difference was checked by performing a t-test ($statistic = -10.05, p - value = 1.888 \times 10^{-23}$), indicating H-H interactions result in a richer expression lexicon than H-A interactions. The EV values were much lower than those reported for negotiation dialogues [6], which may be due to dialogue type: routinisation may form a much greater part of negotiation than it does L2 tutoring. Another reason for the low EV in the H-A corpus is that the student cannot establish expressions other than by chance since the Tutorbot corpus is system-initiated and is not designed to align to the student’s responses. Neither the EV of the H-H nor the H-A corpus was statistically greater than the H- H_R and H- A_R baselines, which can be in part attributed to the high proportion of single-token expressions in both corpora, leading to greater likelihood of their existence in the scrambled baseline.

5.4 Expression Repetition

Expression repetition (ER_S) is the main indication of speaker alignment measured. Figure 2 shows the different degrees

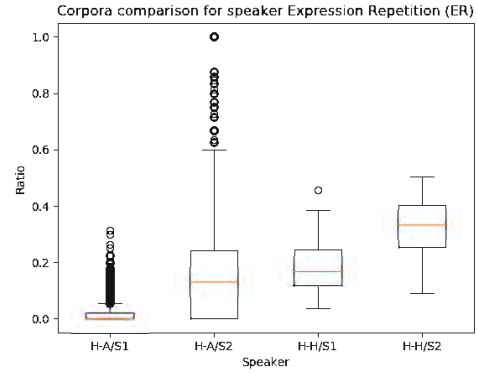


Figure 2: H-H/A corpora ER_S . Speaker difference is significant for H-A ($p < 0.0001$) ($statistic = -44.91, pvalue = 0.0$) and H-H ($p < 0.0001$) ($statistic = -12.71, pvalue = 1.77 \times 10^{-28}$) S1 = Tutor/Agent, S2 = Student

of ERs for both the H-A and H-H corpora. The difference between the ERs of each speaker was significant for both corpora: H-A ($statistic = -44.91, p - value = 0.0$) and H-H ($statistic = -12.71, p - value = 1.770 \times 10^{-28}$). It is interesting to note the asymmetry between speakers for both dialogues. The tutor in the H-H dialogues had a significantly lower proportion of ER than the student, suggesting ER has less to do with teacher strategy as with learner strategy. We compared each ER_S with its ER_R for both corpora: for the H-A corpus, student ER_{S2} ($mean = 0.192, std = 0.235$) was significantly higher than that of $ER_{R,S2}$ ($mean = 0.134, std = 0.206$) ($statistic = -11.20, p - value = 6.593 \times 10^{-29}$). Meanwhile, tutor ER_{S1} ($mean = 0.016, std = 0.032$) was significantly lower than that of their scrambled baseline $ER_{R,S1}$ ($mean = 0.024, std = 0.037$) ($statistic = 9.865, p - value = 8.2012 \times 10^{-23}$) indicating the absence of alignment expected from an agent not designed to do so. For the H-H corpus, student ER_{S2} was not significantly different from their baseline $ER_{R,S2}$ ($statistic = 0.932, p - value = 0.352$), nor was tutor ER_{S1} ($statistic = 2.506, p - value = 0.013$). This can be explained in part by the fact that VO for the H-H corpus ($mean = 0.251, std = 0.061$) was significantly larger than in the H-A corpus ($mean = 0.085, std = 0.146$) ($statistic = -12.32, p - value = 3.089 \times 10^{-34}$).

5.5 Student ER Distribution

In answer to RQ2, we compare the distributions of per-dialogue ER_S values between H-A and H-H corpora. Figure 3 shows histograms of ER frequency for each corpus, which suggest there were multiple types of student alignment in the H-A corpus (a), in contrast to a single cluster of ER values for the H-H corpus (b). To quantify these differences in student alignment – and go beyond a comparison of averages which neglects the possibility of differences across individuals and dialogues – we fit a Bayesian Gaussian Mixture Models [7] (described in Section 4.3) to student ER_S values. The results of our model indicate that the most probable number of clusters, given the data (i.e., the posterior mode), was 5 for the H-A corpus (Figure 3a) and 1 for the H-H corpus (Figure 3b). This analysis also reveals a

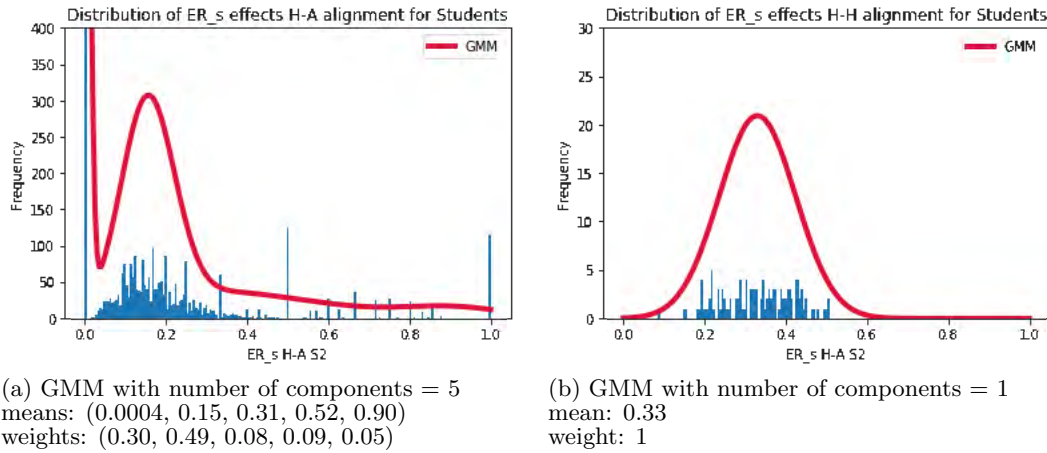


Figure 3: Frequency of Expression Repetition values (High ER indicates greater alignment). Gaussian Mixture Models (GMM) which best fitted to the data shown by the red line. Means: centroids of the component clusters. Weight: the proportion of dialogues in a cluster.

Table 4: Qualitative Analysis of H-A dialogues at the ‘centroids’ of the component clusters

ER	Description and example
0-0.01	no-response or request for help : students either do not engage with the agent, or demonstrate inability to engage
0.1-0.15	minimal response : students respond curtly, appear less engaged <i>bot: What is your favorite day of the week ?</i> <i>user: That 's Sunday .</i>
0.25-0.4	high engagement : dialogues either longer with align and rephrase within longer utterances, without excess repetition, or shorter dialogues consist of more repetition and rephrasing, and the limited vocabulary contributes to alignment <i>bot: Do [you] _have a_ boyfriend or _a_ girlfriend ? Or _a_ husband_ or _a_ wife ?</i> <i>user: I [have [a] husband] .</i>
0.5-0.55	minimal response : low rate of student production, typical response one high-frequency word, low engagement despite high alignment <i>'hi', 'bye.'</i>
0.85-0.9	high repetition : all student responses are rephrases, dialogues very short <i>bot: _Hello_ , _nice to see you_ !</i> <i>user: [Hello] [nice to see you] too</i>

cluster in the H-A corpus which has a qualitatively comparable mean value to the one in H-H (0.310-H-A, 0.330-H-H). Table 4, shows this cluster contains the longest dialogues in Tutorbot, which are qualitatively the most similar to those in BELC.

We hypothesise the other clusters are either, in the case of low level ER, signs of student lack of engagement (alignment being symptomatic of engagement within dialogue) or, in the case of higher ER, signs that the students are in some way conversing in a manner impossible to find in H-H dialogues. We hypothesise either this is due to the communication medium: students can copy, paste and edit the agent utterance to create their response or due to students’ desire to learn through continual repetition of the agent’s phrases.

Table 4 shows examples and descriptions of the H-A corpus data, corresponding to the component means in Figure 3. Since the H-H corpus was gathered as part of an experiment, we know that there would not be ‘outlier’ behaviour present, but the upper and lower ranges show some differences in interaction style of the learner.

6. DISCUSSION

In relation to *RQ1*, whether there is evidence of student - agent alignment in L2 dialogues, we find significant H-A alignment. The magnitude of this effect was weaker than that found in H-H dialogues, and we hypothesise that adaptive student support in the form of tutor alignment is essential for students to align to the degree they do in an L2 H-H setting. We found no significant alignment of agent to student, however an agent designed to interact with more explicit alignment may more resemble the alignment found in the H-H corpus. We found asymmetrical alignment within the H-H corpus, which was in keeping with results reported on lexical priming for the same corpus which found the strongest priming effects are those from student to tutor [17]. In relation to *RQ2*, concerning the exploratory analysis of alignment differences across corpora, a particularly salient finding are the differences in alignment across dialogues, suggesting different patterns of student engagement could be detected via their alignment levels. Table 4 shows that there was a clear ‘normal range’ for interaction, and the outliers showed different signs of student non-engagement. Our key finding is that there was greater variability in H-A compared to H-H alignment (best fit of 5 clusters compared to a single cluster), although role of factors such as dialogue and utterance length in these findings should be investigated in future work. We hypothesise that building a more alignment-focused tutoring agent could increase student engagement and yield results consistent to those within BELC. This could lead to better online L2 tutoring systems which promote student engagement and therefore improve participation and learning. It may be that the nature of an online learning platform will always result in some students who do not fully engage, and need different interven-

tion strategies. Using an alignment metric in the manner of our study could allow for the identification of these students, measurement of their engagement, and prediction of personalised interventions.

7. CONCLUSIONS AND FUTURE WORK

This paper presents a comparative analysis on student to tutor alignment in both an H-A and an H-H dialogue setting. We found students aligned to the agent, although this alignment was not stronger than that present in H-H dialogues which is the case for both negotiation [6] and task-based dialogues [4]. We hypothesise we can better explore this in a setting where the agent is specifically designed to align to the student. A limitation of our study is that both corpora were collected independently and therefore differ in more aspects than the one we wish to explore. In future work it would be desirable to collect data in a controlled setting which is more similar to the Tutorbot corpus to facilitate a more in-depth comparison. Another avenue for future research is the design of adaptive ‘alignment’ moves for the automated tutor to make. The design could draw on how the ZPD influences alignment and what the common ERs are in the H-H corpus, such as confirmatory rephrasing (e.g. “Student: I speak Germanish”, “Tutor: you speak **German**? Great!”) or repetition (e.g. “student: I am 20 years old”, “tutor: **20 years old**? good!”). This research has a number of implications for the educational community, particularly regarding the use of alignment as an indicator of engagement. Furthermore, our method of clustering student ERs to identify ‘normal’ engagement behaviour for a given domain may inform the detection of outliers and has potential for automating dialogue planning and intervention policies.

8. ACKNOWLEDGMENTS

We are grateful for the helpful discussions had with Nicolas Collignon, Edmund Fincham and Pablo Leon and comments from our anonymous reviewers. We thank the team at Babbel and specifically Zach Sporn and Joel Kiesey for making this collaboration possible.

9. REFERENCES

- [1] R. Baker, J. Walonoski, N. Heffernan, I. Roll, A. Corbett, and K. Koedinger. Why students engage in “gaming the system” behavior in interactive learning environments. *Journal of Interactive Learning Research*, 19(2):185–224, 2008.
- [2] S. Bibauw, T. Fran  ois, and P. Desmet. Discussing with a computer to practice a foreign language: research synthesis and conceptual framework of dialogue-based call. *Computer Assisted Language Learning*, 0(0):1–51, 2019.
- [3] C. Biernacki and S. Chr  tien. Degeneracy in the maximum likelihood estimation of univariate gaussian mixtures with em. *Statistics Probability Letters*, 61:373–382, 02 2003.
- [4] H. P. Branigan, M. J. Pickering, J. Pearson, and J. F. McLean. Linguistic alignment between people and computers. *Journal of Pragmatics*, 42(9):2355–2368, 2010.
- [5] A. Costa, M. J. Pickering, and A. Sorace. Alignment in second language dialogue. *Language and cognitive processes*, 23(4):528–556, 2008.
- [6] G. D. Duplessis, C. Clavel, and F. Landragin. Automatic measures to characterise verbal alignment in human-agent interaction. In *18th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 71–81, 2017.
- [7] S. Farrell and S. Lewandowsky. *Computational Modeling of Cognition and Behavior*. 02 2018.
- [8] S. Garrod and M. J. Pickering. Alignment in dialogue. *The Oxford handbook of psycholinguistics*, pages 443–451, 2007.
- [9] R. Hawkes. *Learning to Talk and Talking to Learn: How Spontaneous Teacher-learner Interaction in the Secondary Foreign Languages Classroom Provides Greater Opportunities for L2 Learning*. PhD thesis, University of Cambridge, 2012.
- [10] F. M. Holley and J. K. King. Imitation and correction in foreign language learning. *The Modern Language Journal*, 55(8):494–498, 1971.
- [11] A. Isard, C. Brockmann, and J. Oberlander. Individuality and alignment in generated dialogues. In *Proceedings of the Fourth International Natural Language Generation Conference, INLG ’06*, pages 25–32, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.
- [12] C. Mu  oz. *Age and the rate of foreign language learning*, volume 19. Multilingual Matters, 2006.
- [13] M. J. Pickering and S. Garrod. Toward a mechanistic psychology of dialogue. *Behavioral and Brain Sciences*, 27(2):169–190, 2004.
- [14] D. Reitter and J. D. Moore. Alignment and task success in spoken dialogue. *Journal of Memory and Language*, 76:29–46, 2014.
- [15] P. Robinson and R. Gilabert. Task complexity, the cognition hypothesis and second language learning and performance. *IRAL-International Review of Applied Linguistics in Language Teaching*, 45(3):161–176, 2007.
- [16] A. Sinclair, R. Ferreira, A. Lopez, C. Lucas, and D. Gasevic. I wanna talk like you: Speaker adaptation to dialogue style in l2 practice conversation. In *Proceedings of Artificial Intelligence in Education - 20th International Conference*, 2019.
- [17] A. Sinclair, A. Lopez, C. Lucas, and D. Gasevic. Does ability affect alignment in second language tutorial dialogue? In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 41–50, 2018.
- [18] A. Sinclair, J. Oberlander, and D. Gasevic. Finding the zone of proximal development: Student-tutor second language dialogue interactions. In *Proc. SEMDIAL 2017 (SaarDial) Workshop on the Semantics and Pragmatics of Dialogue*, pages 107–115, 2017.
- [19] L. Vygotsky. Zone of proximal development. *Mind in society: The development of higher psychological processes*, 5291:157, 1987.
- [20] A. Ward and D. Litman. Dialog convergence and learning. *Frontiers in Artificial Intelligence and Applications*, 158:262, 2007.
- [21] A. Ward and D. Litman. Measuring convergence and priming in tutorial dialog. *University of Pittsburgh*, 2007.

Utterance-level Modeling of Indicators of Engaging Classroom Discourse

Cathlyn Stone¹, Patrick J. Donnelly², Meghan Dale³, Sarah Capello³,
Sean Kelly³, Amanda Godley³, Sidney K. D'Mello¹

¹University of Colorado Boulder; ²California State University Chico; ³University of Pittsburgh
430 UCB, Boulder, CO 80309, USA

cathlyn.stone@colorado.edu; pjdonnelly@csuchico.edu; sidney.dmello@colorado.edu

ABSTRACT

We examine the ability of supervised text classification models to identify several discourse properties from teachers' speech with an eye for providing teachers with meaningful automated feedback about the quality of their classroom discourse. We collected audio recordings from 28 teachers from 10 schools in 164 authentic classroom sessions, which we then automatically transcribed into text utterances and then manually coded to identify whether: (1) the utterance contained a question (as opposed to a statement), (2) the question or statement was Instructional vs. Non-Instructional, and (3) the question or statement was Content-Specific. We experimented with Random Forest classifiers and engineered (linguistic, acoustic-prosodic, and contextual) features vs. open-vocabulary n-grams as features to discriminate these discourse variables at the utterance level in a teacher-independent fashion. We achieved AUC scores ranging from 0.71 to 0.77 using open-vocabulary language modeling, which were well above chance (AUC = 0.5), an important step towards our predominant goal of constructing of an automated feedback system for teacher reflection and learning.

Keywords

Automatic Speech Recognition, Natural Language Processing, Dialogic Instruction, Classroom Discourse, Open-vocabulary

1. INTRODUCTION

A teacher's ability to engage students in classroom instruction is of paramount importance in promoting greater student achievement and improving educational outcomes. The level of student engagement is highly dependent upon the ways a teacher interacts with students [1]. The nature of classroom discourse, consisting of the ongoing conversation between the teacher and students, may provide unique insights into a teacher's ability to engage students in the classroom.

Many defining characteristics of classroom discourse have been studied and documented [21]. Traditional methods of classroom instruction are typically presented as monologic discourse, usually in the form of lecture, recitation, and seatwork [20]. However, substantive engagement requires more than just passive listening from students; rather, it requires a degree of student involvement. Not surprisingly, the degree to which classroom

discourse is monologic vs. dialogic has been found to greatly influence student engagement, with higher ratios of student talk providing a necessary condition for improved engagement and dialogic interaction [15, 22]. In order to achieve more widespread classroom engagement, students must not only take notes or listen attentively to well-rehearsed lectures from an instructor but must also be engaged in meaningful conversations about a topic. This deep discussion, a hallmark of dialogic instruction [21], is characterized by a symmetrical balance between student and teacher speech, with social interaction shaping the instruction.

In addition to the ratio of teacher speech to student speech, other trends help to characterize dialogic instruction. For example, teachers who ask more questions tend to promote increased student interaction and discussion in classroom discourse [17]. To this note, the Measures of Effective Teaching Study (MET) found question-asking behavior to be a primary factor in the variability of teaching quality [13]. However, questions are not all created equally. Questions associated with classroom management (like attendance-taking or rhetorical questions which require no student response) are not expected to influence student engagement. Compared to informational questions with a known answer, questions which elicit open-ended responses from students are expected to promote increased levels of engagement [34]. These open-ended, or "authentic" questions, draw upon students' ability to put forth independent thought in forming a response, rather than simply perform an affirmation check of the right answer. Authentic questions also serve to initiate discussion in which students can more thoroughly explore an idea and consider different viewpoints [20]. This in turn helps improve their overall understanding and can increase interest in the subject [21].

Additional defining characteristics of dialog associated with increased engagement, and consequently achievement, include higher levels of uptake (teacher questions which incorporate student responses) and cognitive level (the level of cognitive functioning a teacher question seeks to elicit), among other factors [20]. The study by Gamoran and Kelly (2003) demonstrates the benefits of discussion-based approaches to classroom instruction, which contributed the most towards enhanced student performance on complex literacy [12].

Despite the positive correlation of indicators of dialogic instruction (such as authenticity, uptake, and cognitive level) with increased student engagement and achievement [22], the practice has not gained widespread adoption among teachers. Instead, traditional means of instruction and monologic discourse tend to be most prevalent in the classroom [29]. This might be attributed to the challenges encountered by teachers in adopting sustained dialogic discourse into their pedagogical practices. Most importantly, receiving and learning from feedback is essential to assess current abilities and identify areas for improvement. We know that providing teachers with training and data-driven

Cathlyn Stone, Patrick Donnelly, Meghan Dale, Sarah Capello, Sean Kelly, Amanda Godley and Sidney K. D'Mello "Utterance-level Modeling of Indicators of Engaging Classroom Discourse" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 420 - 425

analysis about their discourse has been shown to positively correlate with student achievement [16]. However, assessment of teachers' instruction via live classroom observations often provides evaluative rather than formative feedback [16]. Moreover, conducting these observations can be expensive, as they require skilled human judges, rubrics, training, and continuous assessment of observers [2]. Therefore, classroom observations occur infrequently, if at all, and must be augmented by additional approaches to further facilitate teacher improvement.

To address this challenge, our study derives from a larger multi-disciplinary project that aims to address a critical lack of quantitative and actionable feedback that teachers receive about the quality of their speech by providing an approach for the automatic analysis of teacher discourse. The present study works towards this overarching goal by automatically classifying teacher utterances from audio recordings of live classroom sessions.

1.1 Related Work

The study of automatic analysis of educational discourse has focused on areas such as online discussions [19], dialog-based intelligent tutoring systems [25], and cognitive models of student learning [5]. In this work we model teachers' classroom discourse through a fully automated process. We examine some of the recent findings in this area.

1.1.1 Modeling classroom discourse

Instructional segment (activity) classification. An instructional segment (or activity) provides coarse-grained information regarding what is occurring at the moment. For example, are students quietly doing seatwork, or is the classroom participating in a discussion or question and answer session? Wang et al. [28] investigated the use of automatic speech recognition of classroom discourse in order to provide feedback for teachers. The authors applied the Language Environment Analysis (LENA) system [11] to segment classroom recordings into broad categories (lecture, discussion, group work). Although the system could provide feedback about the ratio of student to teacher speech, it did not provide any qualitative information about the content or utility of the speech itself. Donnelly et al. [8] also examined automatic identification of instructional activities from classroom recording. Recordings of teacher speech were segmented into individual utterances and transcribed using automatic speech recognition. Using models trained on temporal, natural language, and acoustic features, the authors trained models to identify the dominant (76%) activities (question and answer, procedures and directions, supervised seatwork, group work, and lecture) with accuracies that easily outperformed chance baselines.

Utterance-level classification. At a finer grain than Instructional segment classification, Donnelly et al. [9] built on work by [3] to identify teacher questions within individual utterances. Classroom recordings were segmented, transcribed using Automatic Speech Recognizers (ASRs), and 218 acoustic, linguistic, and contextual features were derived. The acoustic features derived from 384 prosodic, spectral, and voice quality features extracted with the OpenSmile toolkit [10]. Then, a smaller set of 168 acoustic features was obtained by eliminating features with high multicollinearity using tolerance analysis. The transcriptions were analyzed for part-of-speech tags and the presence of specific words (e.g., *why*, *how*) to provide 37 linguistic features. A total of 13 contextual features included timing information, such as the duration of the utterance, the position of the utterance within the

class session, and the duration of the pauses preceding and following the utterance. The authors found that combination of all three modalities made no improvement over linguistic features alone in the task of question identification but did yield small improvements in non-question detection.

Session level classification. Given the positive association between the use of authentic questions and student engagement and achievement [1, 20, 21], any system seeking to provide automatic feedback needs to be able to automatically identify this variable. However, the infrequent use of authentic questions compared to other types of dialog leads to highly imbalanced class distributions that make classification tasks difficult [14]. For this reason, Olney et al. [23] aimed to detect the proportion of authentic questions over the course of the class session, rather than seek to classify individual utterances. Using a model trained on word, part-of-speech, syntactic, and discourse features, the prediction of class-level proportions ($r = 0.50$) outperformed aggregated utterance-level classification ($r = 0.27$), and these results were consistent across low and high dialogic classrooms, and on both ASR and human transcripts. In a follow-up study, Cook et al. [7] compared closed- and open-vocabulary techniques (described in Section 1.1.2) for the same task and found that both approaches were equally predictive of authenticity, but that averaging the models' predictions yielded significant additional improvements.

1.1.2 Computational techniques

We apply techniques from natural language processing and machine learning in the automatic analysis of teacher discourse.

Open-vocabulary language modeling. In contrast to hand-crafted feature sets, open-vocabulary language modeling dynamically generates features for machine learning models by obtaining counts of consecutive words (n-grams) extracted directly from the input text [27]. This "bag-of-n-grams" model then assigns each n-gram feature a value based on its frequency within each utterance. This approach lends itself to human-interpretable analysis of models (e.g., word clouds). Several hyperparameters might also guide the selection of n-grams derived from the training set that should be included in the set of features, as certain n-grams may be unimportant for models, such as n-grams that occur infrequently. These hyperparameters might include whether or not stopwords are removed from text, whether stemming is performed on words, and the types of n-grams considered (unigrams, bigrams, trigrams, etc.). In addition, pointwise mutual information (PMI), described in detail in [6], can be specified as a hyperparameter to filter n-grams by incorporating information about the collocations of words. The PMI for a given n-gram can be defined as $\text{pmi}(n\text{-gram}) = \log(p(n\text{-gram}) / \prod p(\text{word}))$ where $p(n\text{-gram})$ is the probability of an n-gram based on its relative frequency in the training data and $\prod p(\text{word})$ is the product of the probabilities of each word in the n-gram in the training data. This can help ensure only meaningful phrases (such as "*high school*") are used as features. Minimum document frequency might further guide the selection of n-grams (i.e. the n-gram must occur in a specified minimum percentage of all documents to be considered a feature). An overview of these techniques can be found in [6].

1.2 Novelty and Contribution

We apply natural language processing and supervised classification techniques for the *utterance-level* classification of multiple aspects of classroom discourse with an eye for providing

automated feedback to teachers. Our study is novel in multiple respects. First, with the exception of work on detecting individual questions, as noted in Section 1.1.1, existing analyses on the automatic classification of classroom discourse have focused on coarse-grained temporal information ranging from a few minutes to an entire class session. We hypothesize that fine-grained utterance-level information is needed in order to provide meaningful and actionable feedback. Therefore, this study analyzes classroom discourse at the utterance level.

The intrinsic value of a system for automated feedback to teachers is inherently dependent upon the ability to correctly classify *different types* of discourse from automatically segmented recorded speech. Whereas previous work has focused on identifying questions, the present approach considers several more specific discourse variables, which have not previously been studied using automatic recognition methods. In addition to question prediction, we also predict Instructional Questions and Statements as well as Content-Specific Questions and Statements. These discourse variables are described in detail in Section 2.1.2.

2. METHODS

2.1 Dataset

2.1.1 Data collection

Our dataset consists 167 recordings of class sessions, drawn from two sources. One source of data was collected in 2018 and consists of 127 observations from 16 teachers at three schools in western Pennsylvania. Additionally, we newly recoded a subset of the CLASS5 dataset, collected at seven schools in rural Wisconsin over 2014 to 2016. This source of data consists of 40 class observations from 11 teachers [8]. Teachers wore a wireless Samson AirLine 77 vocal headset which transmitted audio to a receiver to then be recorded on a laptop.

2.1.1 Utterance transcription using ASRs

The IBM Watson ASR [26] was used to automatically segment the class recordings into utterances based on hesitations in the audio stream, and to transcribe each resulting utterance. To evaluate the efficacy of the ASR, a sample of 20 utterances per class session was manually transcribed by human coders and these transcriptions were compared to the ASR transcriptions. The average word accuracy (W_{acc}) of the automatic transcriptions across class sessions was 0.602 when considering all utterances. The W_{acc} increased to 0.754 when considering only longer utterances that contained three or more words.

2.1.2 Coding of utterances and coding scheme

To prepare a labeled dataset for training supervised models, a subset of the transcribed utterances was selected for manual annotation by trained human coders. First, to generate this set, any two consecutive utterances were merged together if the pause between them was less than 1.0 seconds. This preprocessing step helped adjoin related phrases together and reduced the number of single word utterances. Next, 200 consecutive merged-utterances were randomly sampled from each class session. If a class session contained less than 200 utterances, then all utterances were sampled for that session. English and language arts content experts trained in the coding schema were given audio excerpts for each utterance in the sampled dataset. The coders manually annotated each utterance with several markers of classroom discourse. Because many of the annotated categories occur only infrequently in the dataset, some markers have been aggregated

together to form binary labels, such as Question or Non-Question. Below we describe the variables used in the current work.

Question/Statement/Fragment. Utterances were coded to determine whether they consisted of a question, statement, or fragment. Questions are defined as requests for information, while conversely, statements are utterances which do not request information. Rhetorical questions, such as, “*It’s the characteristic of a person, right?*” are not coded as questions because they are not requests for information. Fragments are a single word or a few words that have been separated from a cohesive statement or question in the ASR transcription and appear as an individual utterance. Fragments by themselves are meaningless, and it would not be useful to code their discourse properties. To perform binary classification, we combined statements and fragments to predict whether each utterance was a Question or Non-Question.

Instructional questions. Utterances identified as questions were further coded as Instructional or Non-Instructional Questions. Instructional Questions relate to the lesson and its learning goals, whereas Non-Instructional Questions are irrelevant to the lesson and its learning goals, such as questions about student movement and behaviors. For example, “*Who can tell me what a plot diagram is?*” would be coded as an Instructional Question, while “*Why are you late?*” would be considered Non-Instructional.

Instructional question type. Instructional Questions were further coded as Content-Specific, Generic, or Clarifying. Content-Specific Questions inquire about the content/disciplinary practices of the lesson and its learning goals, such as “*What is the theme of the poem?*”, or “*How do you typically revise?*”. Generic Instructional Questions are broad questions about organization, materials, behaviors, or checks for understanding connected to the lesson. Examples include “*Where is your paper from yesterday?*” and “*Does that answer your question?*”. Clarifying Questions are requests for restatements and repetitions, such as “*Can you say that again?*”. We combined the Generic and Clarifying codes to predict the binary classification of Content-Specific Questions vs. Non-Content-Specific Questions.

Instructional statements. Similar to Instructional Questions, utterances identified as Statements were coded as Instructional or Non-Instructional. Instructional Statements relate to the lesson and its learning goals, such as “*A character that moves the action forward but is not central to the story is a minor character*” and “*Today we are going to review literary terms that will be on the quiz on Thursday.*” Non-Instructional Statements are irrelevant to the lesson and its learning goals, such as statements about student movement and behaviors. For instance, “*You shouldn’t be walking around the room. Please sit down.*” In addition, short, placeholder utterances that connote continued thinking (e.g., “*hmm*”, “*um*”, “*okay*”) were coded as Non-Instructional; however, “*okay*” was not automatically coded as Non-Instructional as it can also be an evaluation of a student’s response or serve another function, depending on its context.

Instructional statement type. Instructional Statements were further coded as Content-Specific, Generic, or Reading Aloud. Content-Specific Statements are statements about the content/disciplinary practices of the lesson and learning goals. For example, “*The mood of the play contributes to our understanding of the theme of the play.*” Generic statements are broad statements about organization, behaviors, materials, or checks for understanding connected to the lesson, as in “*Take out your journals, and turn to a new page.*” Reading Aloud statements

occur when the teacher or the students are reading aloud from a text verbatim. If the teacher is reading a short Instructional Statement or discussion question out of a textbook, off a PowerPoint slide, off a worksheet, etc., it is not considered Reading Aloud and is coded as Content-Specific. Furthermore, if the teacher stops reading to make a comment or interjects while the students are reading, those utterances are not coded as Reading Aloud. Similar to predictions made for Instructional Question Type, Generic and Clarifying codes were combined to predict Content-Specific Statements vs. Non-Content-Specific Statements.

2.1.3 Prevalence of discourse types

Our dataset contained a total of 24,755 teacher utterances, with 16,977 from the new Spring 2018 data and 7778 from CLASS5. Table 1 provides information about the prevalence of each of these types of discourse variables in this combined dataset.

Table 1: Summary of dataset

	Count	Proportion
Teacher	24755	
Question	7792	0.31
Instructional	7267	0.29
Content-Specific	5327	0.22
Non-Content-Specific	1940	0.08
Non-Instructional	525	0.02
Non-Question	16963	0.69
Instructional	12113	0.49
Content-Specific	8369	0.34
Non-Content-Specific	3744	0.15
Non-Instructional	4850	0.20

2.2 Machine learning

Using several modalities of features, we trained Random Forest classifiers implemented using the scikit-learn library [24] to perform a binary (present vs. absent) classification of these discourse features. We constructed models using three representations of the input data: as a set of engineered features computed from the audio and transcribed text of utterances, as a set of features derived via open-vocabulary language modeling, and finally as a combination of both of these sources.

We generated the set of engineered features using the acoustic, context, and linguistic features as described in [9]. Acoustic features were extracted from the audio of utterances using the OpenSmile toolkit [10], using the feature set from the 2009 Interspeech Emotion Challenge. This resulted in 384 acoustic features. Context features describe properties of the utterance such as its duration, its normalized (to unit variance) position in the overall classroom session, and the length of time of the surrounding pauses. In total, we considered 13 context features. Linguistic analyzers parsed the transcribed text and identified the presence of known question words and part of speech tags. These were found using the Brill Tagger [4] to identify certain question words, part-of-speech tags, and other keywords, resulting in 37 total features. The values of all these features were standardized to have a mean of 0 and unit variance. Standardization was computed using the formula $z = (x-u) / s$ such that z is the standardized score, x is the value of an individual sample, u the mean value of all training samples, and s is the standard deviation of the samples.

A bag-of-n-grams representation of input formed the open-vocabulary feature set. N-grams (of which we considered unigrams, bigrams, and trigrams) derived from the texts of

transcribed utterances were filtered according to the values of a few hyperparameters. We experimented using minimum document frequencies of 0.01, 0.02, and 0.03; PMI values of 0.2 and 0.4; and either including or excluding stopwords (see Section 1.1.2).

We implemented teacher-level 5-fold cross-validation to determine the best set of hyperparameters for models within each training fold. Specifically, we ensured that all utterances from the same teacher were always kept within the same train/test/validation fold. This helps ensure generalizability of our approach to new data and new teachers. To enable faster training of models, we limited the overall search space of hyperparameters, varying the parameters specified for the open-vocabulary models and leaving other parameters at default values as specified by scikit-learn. To overcome the underlying class imbalance in the dataset (see Table 1), we experimented using the imblearn library [18] to resample the minority class utterances such that both classes were more equally represented in the input dataset. This approach was applied to all models and only performed on the training set; class distributions in the validation and testing sets were unchanged.

3. RESULTS

We examined the ability of different types of models to predict five indicators of teacher discourse using utterances automatically segmented and transcribed by an ASR. We used area under the receiver operating characteristic curve (AUROC or AUC) as our primary outcome metric, which we computed using the pooled predicted probabilities from the five folds of our dataset. An AUC of 0.5 would signify chance performance.

3.1 Predictive language features

Table 2. Top 10 correlated n-grams

Variable	Top 10 correlated n-grams	Example sentences from dataset
Question	<i>does, did, think, good, say, mean, yes, guys, kind, make</i>	“why do you <i>say</i> you want to do”
Instructional Question	<i>does, did, think, good, say, yes, kind, mean, guys, make</i>	“are you <i>guys</i> doing”; “ <i>did</i> you talk to me on Friday”
Content-Specific Question	<i>does, think, did, good, kind, say, know, make, mean, people</i>	“okay why <i>does</i> she <i>think</i> it’s any better for her son”; “what <i>does</i> that <i>mean</i> ”
Instructional Statement	<i>na, gon, gon na, going, just, like, right, <hesitation>, look, little</i>	“all <i>right</i> now notice what you need to do <i>look</i> at this part”
Content-Specific Statement	<i>like, <hesitation>, going, na, just, gonna, gon, kind, little, right</i>	“ <i>like</i> if I’d done that all <i>right</i> I have a sample body paragraph here”

Note: <hesitation> expresses a token generated by ASRs to indicate hesitation in speech. Here we treat it as a word.

We analyzed our models to correlate the top 10 n-grams for each discourse variable in order to investigate characteristic language features. We calculated Spearman correlations of n-grams to the class labels (either 0 or 1) of the documents in which they appear.

These correlations were averaged across the five folds on which Random Forest models were trained. These n-grams are listed in Table 2, and we note some expected patterns. For example, one would expect that questions would be characterized by auxiliary verbs such as *does* and *did* as well as action verbs such as *think*, *say*, and *make*. We also observed considerable overlap between these categories. For example, both Instructional and Content-Specific questions include *does*, *think*, and *did* among their top three n-grams and share eight of ten most common n-grams. Likewise, Instructional and Content-Specific Statements share nine of the top ten most common n-grams. Conversely, we found that Content-Specific Statements and Questions have overlap only in the n-gram *kind*.

3.2 Comparison of feature sets

We constructed Random Forest models using three types of features: (1) engineered acoustic, context, and linguistic features, (2) bag-of-n-grams language features via open-vocabulary language modeling, and (3) a combination of both. Results using the Random Forest model are shown in Figure 1. We found that open-vocabulary language modeling resulted in the highest average AUC scores (average AUC = 0.74) for all discourse variables, followed by the combined set of features (average AUC = 0.72), while the engineered features were the least predictive (average AUC = 0.68). With respect to question detection, for which we have a baseline from previous work [9], we found that the current approach with language features yielded a 11% improvement over engineered features alone. These results demonstrate significant improvement (3-12%) over the previous state of the art.

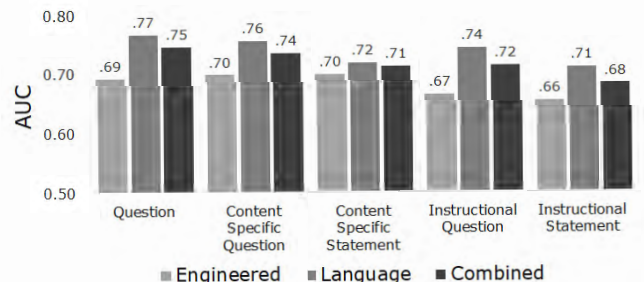


Figure 1. Random Forest: AUROC per feature set

4. DISCUSSION

We investigated the extent to which several characteristics of classrooms discourse could be automatically identified at the utterance level. While prior work focused on predicting questions at the utterance level, in this study we detected several additional discourse characteristics at the utterance level, which would be of paramount importance in a real-time feedback system for teachers. We collected additional data from live classroom sessions during the Spring of 2018 to augment previously collected data. We developed a new coding scheme and manually annotated the dataset. Audio recordings of entire classroom sessions were automatically segmented into utterances which were then manually coded by humans and transcribed into text by the IBM Watson ASR. We then executed machine learning experiments to examine the extent to which these discourse variables could be recognized from the audio signal alone.

4.1 Main findings

First, we observed that open-vocabulary bag-of-n-grams Random Forest models outperformed our previous attempt using models built using only engineered features. These results demonstrate that the specific words a teacher uses, as determined by automatic transcriptions, may be of more utility than acoustic and prosodic cues, timing cues, rates of speech, parts-of-speech analysis, and closed-vocabulary word lists. Moreover, these findings indicate that the words most useful to differentiate between dialogic acts often differ from those anticipated by domain-specific closed-vocabulary lists. For example, closed-vocabulary lists created to predict questions may only look for whose words typically indicative of questions (e.g., *what*, *where*, *why*, *how*), while overlooking other words that may also be useful to distinguish this type of discourse (such as *think*, *say*, *mean*).

In summary, our results using open-vocabulary modeling (with AUCs ranging from 0.71 to 0.77) comfortably outperformed chance (AUC = 0.5), and reflect the state of the art performance on automatic modeling of classroom discourse. Further, the fact that the models were trained in a manner that generalizes to new teachers, and that the training data included audio from two different U.S. states across varying grade levels (mainly middle school for CLASS 5 vs. mainly high school for Spring 2018 data), increases our confidence in their generalizability. As such, we are optimistic that our present results reflect the feasibility of fully-automated utterance-level classrooms discourse modeling, a key step towards providing actionable feedback for teachers.

4.2 Limitations and future work

Although research indicates that the dialogic indicators of authenticity, uptake, and cognitive level are predictors of enhanced student engagement, this study does not aim to identify these indicators at the utterance level. This is because these variables have extremely low base rates (all under 10%), resulting in severe class imbalance when attempting to identify them from *all* teacher utterances. However, the automatic recognition of the discourse variables in this study serves as a precursor for subsequent approaches to better accurately identify these useful but infrequently occurring dialogic variables. The identification of these key dialogic variables relies on the ability to first correctly differentiate between more generic discourse properties, such as Questions vs. Statements, followed by Content-Specific Questions (of which Authentic Questions are a subset) vs. Instructional Questions.

In addition, we are currently limited by the lack of annotated data to provide sufficient exemplars of these specific dialogic properties (authenticity, uptake, and cognitive level) at the utterance level. Thus, additional collection of data would allow more examples of these more rarely occurring discourse types. Given this new data, we will extend our models to attempt to identify these infrequently occurring dialogic indicators.

Furthermore, continued improvement in the accuracy of our predictions is necessary to ensure the value of the assessment and feedback from our automated system. We plan on exploring several improvements to advance this goal. First, we will incorporate transcription metadata, such as the confidence values of the ASRs, in the models in order to weight individual words in the open-language model based on the quality of the transcription. Since words transcribed with a low confidence may be misidentified, excluding or discounting these words from language model may help to reduce modeling error. Second, we

will empirically experiment with varying the pause threshold used for segmentation. Perhaps a slightly longer or shorter gap in speech would provide a better separator of utterances. Third, we will continue to explore different supervised machine learning models or neural network architectures to further improve our ability to automatically identify these discourse indicators.

4.3 Concluding remarks

We hope that in our continued efforts towards automatic prediction of types of discourse, we can achieve the capability to provide valuable, actionable feedback to teachers about their instructional techniques so that they can better engage students in learning. Certainly, much work remains to be done in this area in order to improve upon our current ability. Nonetheless, this study forms an important step towards our overarching goal and serves as a foundation for future work in this area.

REFERENCES

- [1] Applebee, A.N. et al. 2003. Discussion-based approaches to developing understanding: Classroom instruction and student performance in middle and high school English. *American Educational Research Journal*. 40, 3 (2003), 685–730.
- [2] Archer, J. et al. 2016. *Better feedback for better teaching: A practical guide to improving classroom observations*. John Wiley & Sons.
- [3] Blanchard, N. et al. 2016. Automatic detection of teacher questions from audio in live classrooms. *Proceedings of the 9th International Conference on Educational Data Mining (EDM 2016)*, International Educational Data Mining Society (2016), 288–291.
- [4] Brill, E. 1992. A simple rule-based part of speech tagger. In *Proceedings of the workshop on Speech and Natural Language* (1992), 112–116.
- [5] Chen, S.Y. and Macredie, R.D. 2004. Cognitive modeling of student learning in web-based instructional programs. *International Journal of Human-Computer Interaction*. 17, 3 (2004), 375–402.
- [6] Church, K.W. and Hanks, P. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*. 16, 1 (1990), 22–29.
- [7] Cook, C. et al. 2018. An Open Vocabulary Approach for Estimating Teacher Use of Authentic Questions in Classroom Discourse. *11th International Conference on Educational Data Mining* (2018), 116–126.
- [8] Donnelly, P.J. et al. 2016. Automatic teacher modeling from live classroom audio. *Proceedings of the 24th Conference on User Modeling, Adaptation and Personalization (UMAP 2016)*. (2016), 45–53.
- [9] Donnelly, P.J. et al. 2017. Words matter: Automatic detection of questions in classroom discourse using linguistics, paralinguistics, and context. *Lak '17: Proceedings of the seventh international conference on learning analytics & knowledge*. (2017), 218–227.
- [10] Eyben, F. et al. 2010. OpenSmile: the Munich versatile and fast open-source audio feature extractor. *Proceedings of the 18th ACM international conference on Multimedia* (2010), 1459–1462.
- [11] Ford, M. et al. 2008. *The LENA Language Environment Analysis System*. Technical Report LTR-03-2. Boulder, CO: LENA Foundation.
- [12] Gamoran, A. and Kelly, S. 2003. Tracking, instruction, and unequal literacy in secondary school English. *Stability and change in American education: Structure, process, and outcomes*. (2003), 109–126.
- [13] Kane, T. and Cantrell, S. 2010. Learning about teaching: Initial findings from the measures of effective teaching project. *MET Project Research Paper, Bill & Melinda Gates Foundation*. 9, (2010).
- [14] Kelly, S. et al. 2018. Automatically Measuring Question Authenticity in Real-World Classrooms. *Educational Researcher*. 47, 7 (2018), 451–464.
- [15] Kelly, S. 2007. Classroom discourse and the distribution of student engagement. *Social Psychology of Education*. 10, 3 (2007), 331–352.
- [16] Lai, M.K. and McNaughton, S. 2013. Analysis and discussion of classroom and achievement data to raise student achievement. *Data-based decision making in education*. Springer. 23–47.
- [17] Lee, Y. and Kinzie, M.B. 2012. Teacher question and student response with regard to cognition and language use. *Instructional Science*. 40, 6 (2012), 857–874.
- [18] Lemaître, G. et al. 2017. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*. 18, 1 (2017), 559–563.
- [19] Mu, J. et al. 2012. The ACODEA framework: Developing segmentation and classification schemes for fully automatic analysis of online discussions. *International Journal of Computer-Supported Collaborative Learning*. 7, 2 (2012), 285–305.
- [20] Nystrand, M. et al. 1997. *Opening dialogue: Understanding the dynamics of language and learning in the English classroom*. Language and Literacy Series.
- [21] Nystrand, M. et al. 2003. Questions in time: Investigating the structure and dynamics of unfolding classroom discourse. *Discourse processes*. 35, 2 (2003), 135–198.
- [22] Nystrand, M. and Gamoran, A. 1991. Instructional discourse, student engagement, and literature achievement. *Research in the Teaching of English*. (1991), 261–290.
- [23] Olney, A. et al. 2017. Assessing the Dialogic Properties of Classroom Discourse: Proportion Models for Imbalanced Classes. *Proceedings of the 10th International Conference on Educational Data Mining*. (2017), 162–167.
- [24] Pedregosa, F. et al. 2011. Scikit-learn: Machine Learning in Python. In *Journal of Machine Learning Research*. (2011), 2825–2830.
- [25] Rus, V. et al. 2013. Recent advances in conversational intelligent tutoring systems. *AI magazine*. 34, 3 (2013), 42–54.
- [26] Saon, G. et al. 2015. The IBM 2015 English conversational telephone speech recognition system. In *Proc. Interspeech*. (2015), 3140–3144.
- [27] Tan, C.-M. et al. 2002. The use of bigrams to enhance text categorization. *Information processing & management*. 38, 4 (2002), 529–546.
- [28] Wang, Z. et al. 2013. Using the LENA in teacher training: Promoting student involvement through automated feedback. *Unterrichtswissenschaft*. 4, (2013), 290–305.
- [29] Wells, G. and Arauz, R.M. 2006. Dialogue in the classroom. *The journal of the learning sciences*. 15, 3 (2006), 379–428.

Probabilistic Modeling of Peer Correction and Peer Assessment

Takeru Sunahase
Kyoto University
stakeru@ml.ist.i.kyoto-u.ac.jp

Yukino Baba
University of Tsukuba
baba@cs.tsukuba.ac.jp

Hisashi Kashima
Kyoto University;
RIKEN Center for AIP
kashima@i.kyoto-u.ac.jp

ABSTRACT

Peer assessment is a promising solution for scaling up the grading of a large number of submissions. The reliability of evaluations is one of the critical issues in peer assessment; several probabilistic models have been proposed for obtaining reliable grades from peers. *Peer correction* is a similar framework, in which students are instructed to correct the errors in submissions from other students. Peer correction is typically performed simultaneously with peer assessment; a reviewer is instructed to correct the errors in a submission and to provide a grade to it. We observe the occasional inconsistency between a grade and the correction; for example, a reviewer provides a high grade for a submission but she corrects many errors in it. Such inconsistencies can point to unreliable reviewers. In this paper, we propose probabilistic models for peer correction, and the combination of the peer correction models and the existing peer assessment models for capturing the inconsistency to accurately estimate the reviewer reliability and the student ability. We conduct experiments using the dataset of an actual peer correction platform for language translation, and the results demonstrate that the combination of peer correction models and peer assessment models improves the accuracy of the student ability estimation.

Keywords

Peer correction, peer assessment, statistical models

1. INTRODUCTION

MOOCs have changed education by offering open access to university course materials; however, not everything performed in offline classes is effectively introduced in MOOCs. An example is the ability assessment; in offline classes, teachers evaluate the student abilities by examining their submitted assignments and decide how to improve the educational efficiency. In contrast, assessing the abilities of tens of thousands of students in MOOCs is not feasible for teachers.

A promising solution for large-scale ability assessment is to allow students themselves to be involved in the evaluation; instead of teachers, students grade the submissions from other students. Such *peer assessment* approach is beneficial for scaling-up the ability assessment and it has been applied to several MOOCs courses [7]. However, the reliability of evaluations is one of the critical issues in peer assessment because some students may provide unreliable evaluations owing to laziness or lack of evaluation skills. Several probabilistic models have been proposed for estimating the reliabilities of the reviewers in order to accurately assess the student abilities in peer assessment [7, 4, 11, 8, 13, 6]. These models are based on the assumption that students with high ability are likely to provide reliable grades. The models are used to estimate the ability of a student as a test taker and the reliability as a reviewer.

In a similar framework of peer assessment, called *peer correction*, students correct the errors in the submissions from other students. Peer correction is helpful for teachers to reduce their efforts for providing feedback to the students. Typically, peer correction is performed simultaneously with peer assessment; a student is instructed to grade a submission and to correct its errors.

Although the outcomes of peer correction are naturally assumed to be informative for estimating the student abilities, probabilistic models for peer correction have not yet been investigated. Based on a natural assumption that a student who receives fewer corrections are likely to have a higher ability, we propose probabilistic models for peer correction that capture the relationship between the student abilities and the correction outcomes.

Additionally, we noticed an inconsistency between the outcomes of peer correction and those of peer assessment. In one case, a reviewer provides a high grade to a submission but she corrects many errors in it; in another case, a reviewer assigns a low grade but she does not make any corrections. Our idea is that such inconsistencies are beneficial in determining unreliable reviewers; thus, we propose to combine peer assessment models with our peer correction models. This combination allows us to capture the inconsistency and to incorporate it into the estimation of the reviewer reliability and the student ability.

We conduct experiments using a peer correction dataset about language translation. The results of the experiments

Takeru Sunahase, Yukino Baba and Hisashi Kashima
"Probabilistic Modeling of Peer Correction and Peer Assessment"
In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 426 - 431

show that our probabilistic models for peer correction are capable of estimating the student abilities, and the combined models of peer correction and peer assessment demonstrate a better performance in determining high-ability students than the peer assessment models.

The contributions of this paper are twofold: (i) we propose novel probabilistic models for peer correction that enable us to estimate the student abilities from the received corrections (Section 4), and (ii) we propose to combine our peer correction models and peer assessment models to exploit the inconsistencies among the outcomes of corrections and assessments (Section 6); the results of the experiments show that the combined models are efficient in accurately estimating the student abilities.

2. PROBLEM DEFINITION

We begin with the formulation of peer assessment and peer correction. We assume there is a set of students \mathcal{S} . When a student creates a submission for an assignment, other students (that we call *reviewers*) evaluate it and assign grades. The grade for the student $u \in \mathcal{S}$ assigned by the reviewer $v \in \mathcal{S}$ is denoted by $z_{uv} \in \mathbb{R}$. Each reviewer is additionally instructed to correct the errors in a submission. A correction result is denoted by y_{uv} . If a reviewer does not provide any correction for a submission, such information is also embedded in y_{uv} . The representation of y_{uv} is discussed in the next section.

Given a set of peer assessment and peer correction outcomes, \mathcal{D} , each of which is represented by a tuple (u, v, z_{uv}, y_{uv}) , our goal is to estimate the true abilities of the students $\{s_u\}_{u \in \mathcal{S}}$, where $s_u \in \mathbb{R}$.

3. DATASET

In this work, we use a peer assessment and peer correction dataset collected from Conyac¹, which is a crowdsourcing language translation platform. This platform employs peer correction and peer assessment between translators for collaboratively improving their skills; thus, a translator on this platform can be considered as a student. When a student submits a translation, other students evaluate its quality on a five-point scale (zero (low) to four (high)) and correct the errors in it. Students are invited to high-reward jobs if they have reviewed several submissions.

Students on Conyac can take a qualification test to demonstrate their skills. On this test, a student is instructed to translate the given sentences and then the translations are evaluated by experts employed by the service provider. According to the score, a student is assigned one of five expertise levels (D, C, B, A, and A+). This level is used for the job assignment and the default level is set to one. We consider the assigned levels as the ground truth of the student abilities, that we aim to estimate from the outcomes of peer assessment and peer correction.

We target the peer assessment and peer correction for Japanese to English translations on Conyac. Our dataset contains 5,008 reviews for 413 students, and 135 students

provide at least one review. Figure 1(a) shows the distribution of the grades assigned to translations and Figure 1(b) illustrates the distribution of the students' true expertise levels.

We conduct exploratory data analysis to investigate how the outcomes of peer correction can be used for estimating student expertise levels. A natural expectation is that a student whose submissions are likely to be corrected would have lower ability. We calculate the correction ratio of each student, which is the number of corrected submissions divided by the number of submissions. Figure 1(c) shows the average correction ratio of the students in each expertise level. We observe that students with the highest level are likely to have lower correction ratios than the others.

Additionally, we consider that students who have more errors in their submissions would be have lower ability. We calculate the number of corrected parts in each submission by applying the Gestalt pattern matching [10]. We first obtain the matched patterns in pre-correction and post-correction submissions, and then count the number of unmatched patterns in the post-correction submissions. The examples of the calculated numbers are shown in Table 1 and Figure 1(d) shows the distribution of the number of corrected parts in each submission. We calculated the average number of corrected parts of each student and Figure 1(e) presents the average of the values at each level. We found that the students with higher levels are likely to have a lower number of corrected parts.

From these observations, we decide to use the following binary and numerical variables to represent a correction outcome: (1) $y_{uv}^{(b)} \in \{0, 1\}$, which indicates whether the corresponding submission is corrected by the grader ($y_{uv}^{(b)} = 0$) or not ($y_{uv}^{(b)} = 1$), (2) $y_{uv}^{(n)} \in \{0, 1, 2, \dots\}$, which indicates the number of parts corrected by the grader.

4. PEER CORRECTION MODELS

We propose two peer correction models, PC_b and PC_n , for estimating the student true abilities. The models are illustrated in Figure 2(a).

4.1 PC_b model

We first present a generative model for $y_{uv}^{(b)} \in \{0, 1\}$, which is a binary indicator whether the submission has been corrected by the reviewer or not. We have two latent parameters into our model, that is, student true ability and reviewer bias; each student is associated with the latent true ability, $s_u \in \mathbb{R}$, which we aim to estimate, and each reviewer has a different bias parameter, $b_v \in \mathbb{R}$, presuming that a reviewer with a lower bias tends to review a submission negatively.

Following the observations, we assume that a submission from a student is likely to be not corrected by a reviewer if the student has high ability. In addition, a reviewer is not likely to correct a submission if he/she has a higher bias. These assumptions are represented as the following generative model:

$$y_{uv}^{(b)} \sim \text{Bern}\left(y_{uv}^{(b)} \mid \sigma(s_u + b_v + r)\right), \quad (1)$$

where $\sigma(x) = 1 / (1 + \exp(-x))$, $\text{Bern}(\cdot)$ is the Bernoulli dis-

¹<https://conyac.cc/>

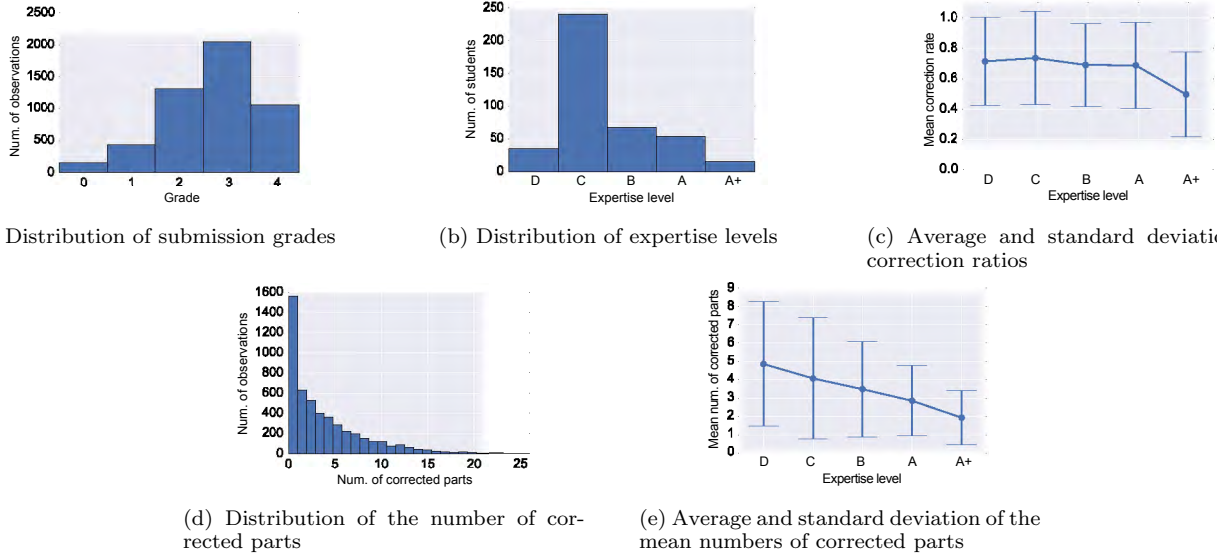


Figure 1: Statistics of our dataset

Table 1: Examples of the calculated number of corrected parts. The corrected parts are highlighted (modified parts are highlighted in yellow, added parts are highlighted in pink, and removed parts are highlighted in blue). There were seven corrected parts in the last example because there were three modified parts (“Current members,” “are” and “was working on the”), two added parts (“female” and “,”), and two removed parts (“girls” and “the”).

Pre-correction	Post-correction	Num. of corrected parts
Please enter the title. Please enter the application conditions.	Please enter the title. Please enter the eligibility requirements.	1
41 people from major travel agencies of Japan and land operators participated and had the business meetings with suppliers about latest Thailand MICE circumstances.	41 people from major travel agencies of Japan and land operators participated and had the business meetings with suppliers about the latest Thailand MICE circumstances.	1
Kalafina is the vocal girls band produced by Yuki Kajiura. Currently the member of Kalafina is WAKANA, KEIKO and HIKARU. The group was formed in order to produce the main song when the composer Yuki Kajiura produced music for the film “Boundary of Emptiness”.	Kalafina is the female vocal band produced by Yuki Kajiura. Current members of Kalafina are WAKANA, KEIKO, and HIKARU. The group was formed in order to produce the main song when composer Yuki Kajiura was working on the music for the film “Boundary of Emptiness”.	7

tribution, and r is a noise. Note that $y_{uv}^{(b)} = 1$ indicates that the corresponding submission is *not* corrected by the reviewer. We denote this generative model by PC_b model. We can interpret $s_u + b_v + r$ as an apparent ability of the student u for the reviewer v at the time. The model indicates that a submission is likely to be not corrected when the apparent ability is high.

In the same way as the existing peer assessment models that will be reviewed in the next section, we use normal distributions as priors for s_u , b_v , and r :

$$\text{(Student ability)} \quad s_u \sim \mathcal{N}(s_u | \mu_0, 1/\gamma_0) \quad (2)$$

$$\text{(Reviewer bias)} \quad b_v \sim \mathcal{N}(b_v | 0, 1/\eta_0) \quad (3)$$

$$\text{(Noise)} \quad r \sim \mathcal{N}(r | 0, 1/\kappa_0), \quad (4)$$

where μ_0 , γ_0 , η_0 , and κ_0 are hyperparameters.

4.2 PC_n model

Our second model targets the number of corrected parts in each correction, $y_{uv}^{(n)} \in \{0, 1, 2, \dots\}$. Following the observations from the actual dataset, we assume that a reviewer corrects more parts of a submission when the student has a lower ability. We use the Poisson distribution to represent this assumption:

$$y_{uv}^{(n)} \sim \text{Poisson} \left(y_{uv}^{(n)} \middle| \frac{1}{\exp(s_u + b_v + r)} \right).$$

Similar to the PC_b model, $s_u + b_v + r$ is considered as the apparent ability of the student u to the reviewer v , and this model indicates that more parts of the submission is likely to be corrected by the reviewer if the apparent ability of the student is lower. We call this model PC_n . The priors given in Eqs. (2), (3), and (4) are incorporated into the PC_n model as well.

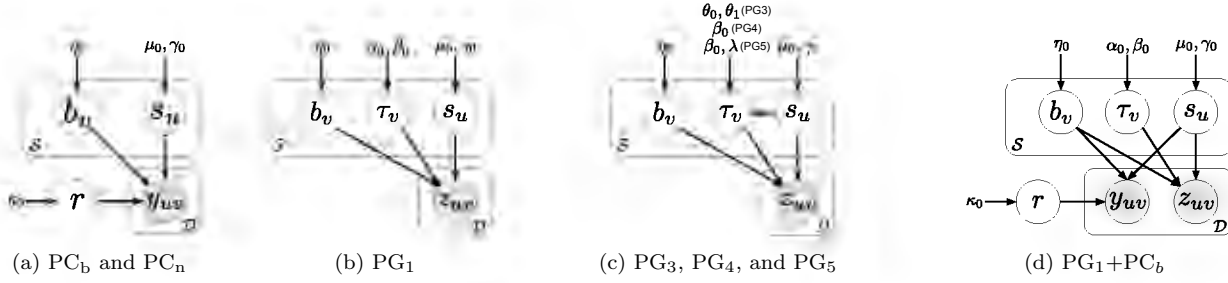


Figure 2: Peer correction and peer assessment models, and combined models

5. PEER ASSESSMENT MODELS

We next review the existing peer assessment models, which are combined with our peer correction models in the next section. In particular, we summarize the PG₁ [7], PG₃ [7], PG₄ [6], and PG₅ [6] models². These are the generative models for grades, $z_{uv} \in \mathbb{R}$. The peer assessment models are illustrated in Figures 2(b) and 2(c).

The student ability and the reviewer bias parameters are also incorporated in the peer assessment models. All the models use the same priors given in Eqs. (2) and (3).

5.1 PG₁ and PG₃ models

In addition to the latent parameters incorporated in the peer correction models (s_u and b_v), the peer assessment models contain the reviewer reliability $\tau_v \in \mathbb{R}^+$. This parameter indicates how likely a grade given by the reviewer contains a noise. PG₁ is defined as follows:

$$\begin{aligned} \text{(Reviewer reliability)} \quad \tau_v &\sim \text{Gamma}(\tau_v | \alpha_0, \beta_0) \\ \text{(Outcome)} \quad z_{uv} &\sim \mathcal{N}(z_{uv} | s_u + b_v, 1/\tau_v), \end{aligned}$$

where α_0 and β_0 are hyper parameters. PG₃ is an extension of PG₁, which incorporates the relationship between the reviewer reliability and the ability of the reviewer (as a student). PG₃ is given as follows:

$$\begin{aligned} \text{(Reviewer reliability)} \quad \tau_v &= \theta_1 s_v + \theta_0 \\ \text{(Outcome)} \quad z_{uv} &\sim \mathcal{N}(z_{uv} | s_u + b_v, 1/\tau_v), \end{aligned}$$

where θ_0 and θ_1 are hyperparameters.

5.2 PG₄ and PG₅ models

PG₄ and PG₅ are variations of PG₃ and they incorporate the relationship between the reviewer reliability and the reviewer ability into the priors of the reliability parameter. The generative models of the reviewer reliability and outcome in PG₄ are given as follows:

$$\begin{aligned} \text{(Reviewer reliability)} \quad \tau_v &\sim \text{Gamma}(\tau_v | s_v, \beta_0) \\ \text{(Outcome)} \quad z_{uv} &\sim \mathcal{N}(z_{uv} | s_u + b_v, 1/\tau_v), \end{aligned}$$

and those in PG₅ are given as follows:

$$\begin{aligned} \text{(Reviewer reliability)} \quad \tau_v &\sim \mathcal{N}(\tau_v | s_v, 1/\beta_0) \\ \text{(Outcome)} \quad z_{uv} &\sim \mathcal{N}(z_{uv} | s_u + b_v, \lambda/\tau_v), \end{aligned}$$

where β_0 and λ are hyperparameters.

²We do not include PG₂ [7], which is almost similar to PG₁ except it incorporates time-series factors.

6. COMBINED MODELS FOR PEER CORRECTION AND PEER ASSESSMENT

We finally combine our peer correction models and the existing peer assessment models. By combining these two types of models, we expect to capture an inconsistency between the outcome of peer correction and that of peer assessment; the inconsistency can be informative for estimating the reviewer reliabilities.

We use PG₁ and PC_b to explain the model combining and we term the combined model as PG₁+PC_b. We simply consider that s_u and b_v are shared between these two models; namely, the generative model for PG₁+PC_b is given as:

$$\begin{aligned} \text{(Student ability)} \quad s_u &\sim \mathcal{N}(s_u | \mu_0, 1/\gamma_0) \\ \text{(Reviewer reliability)} \quad \tau_v &\sim \text{Gamma}(\tau_v | \alpha_0, \beta_0) \\ \text{(Reviewer bias)} \quad b_v &\sim \mathcal{N}(b_v | 0, 1/\eta_0) \\ \text{(Noise)} \quad r &\sim \mathcal{N}(r | 0, 1/\kappa_0) \\ \text{(Outcomes)} \quad z_{uv} &\sim \mathcal{N}(z_{uv} | s_u + b_v, 1/\tau_v), \text{ and} \\ y_{uv}^{(b)} &\sim \text{Bern}(y_{uv}^{(b)} | \sigma(s_u + b_v + r)). \end{aligned}$$

The PG₁+PC_b model is illustrated in Figure 2(d). Other combined models are defined similarly as PG₁+PC_b.

When an inconsistency occurs between corrections and grades, i.e., a reviewer provides a high grade to a submission but makes many corrections in it, we consider that a large noise occurs on the grade (z_{uv}) and thus the reliability of the reviewer (τ_v) is estimated as low. The combination of peer assessment models and peer correction models allows us to leverage such inconsistencies to estimate the reviewer reliabilities and the student abilities.

7. EXPERIMENTS

We conduct experiments using the actual peer assessment and peer correction dataset about language translation. We investigate the effectiveness of the proposed methods to estimate the student abilities.

7.1 Baselines

We compare the proposed models (PC_b, PC_n, and PG_{1,3,4,5}+PC_{b,n}) with the following baselines: **(a) Correction ratio (PC_b[#])**: this is a naïve version of PC_b and considers the correction ratio of each student as the ability. Specifically, the correction ratio is defined as $-\sum_{y_{uv}^{(b)} \in \mathcal{Y}_u^{(b)}} \delta(y_{uv}^{(b)} = 0) / |\mathcal{Y}_u^{(b)}|$, where $\mathcal{Y}_u^{(b)}$ is the set of correction outcomes for the student u , and $\delta(\cdot)$ is the in-

Table 2: Average and standard deviation of AUC scores of each method on various classification boundaries. Each column indicates the results for each classification boundary; for example, (D, C, B, A | A+) represents the results for classifying the students at A+ and the others. The winner for each boundary is bold-faced. The cases where a combined model outperforms the corresponding peer assessment model (PG₁, PG₃, PG₄, or PG₅) are underlined.

	AUC			
	(D, C, B, A A+)	(D, C, B A, A+)	(D, C B, A, A+)	(D C, B, A, A+)
PC _b [#]	0.713 ± 0.020	0.584 ± 0.015	0.580 ± 0.013	0.498 ± 0.025
PC _b	0.805 ± 0.037	0.628 ± 0.008	0.604 ± 0.013	0.483 ± 0.034
PC _n [#]	0.714 ± 0.045	0.627 ± 0.013	0.598 ± 0.010	0.611 ± 0.041
PC _n	0.832 ± 0.050	0.710 ± 0.012	0.690 ± 0.008	0.602 ± 0.046
PG [#]	0.794 ± 0.023	0.723 ± 0.015	0.697 ± 0.013	0.810 ± 0.011
PG ₁	0.845 ± 0.016	0.739 ± 0.025	0.742 ± 0.013	0.801 ± 0.015
PG ₃	0.751 ± 0.193	0.756 ± 0.033	0.725 ± 0.020	0.786 ± 0.020
PG ₄	0.862 ± 0.069	0.774 ± 0.012	0.743 ± 0.015	0.791 ± 0.011
PG ₅	0.842 ± 0.115	0.775 ± 0.029	0.731 ± 0.015	0.759 ± 0.043
PG ₁ +PC _b	0.821 ± 0.020	<u>0.755</u> ± 0.016	0.736 ± 0.010	0.792 ± 0.011
PG ₃ +PC _b	<u>0.841</u> ± 0.137	<u>0.779</u> ± 0.026	0.736 ± 0.011	<u>0.789</u> ± 0.030
PG ₄ +PC _b	<u>0.870</u> ± 0.019	0.764 ± 0.008	0.730 ± 0.021	<u>0.800</u> ± 0.016
PG ₅ +PC _b	0.914 ± 0.018	0.782 ± 0.016	0.719 ± 0.026	<u>0.782</u> ± 0.032
PG ₁ +PC _n	<u>0.846</u> ± 0.019	0.737 ± 0.017	0.726 ± 0.009	0.661 ± 0.047
PG ₃ +PC _n	0.788 ± 0.153	0.705 ± 0.044	0.704 ± 0.022	0.710 ± 0.060
PG ₄ +PC _n	0.844 ± 0.038	0.753 ± 0.017	0.724 ± 0.018	0.646 ± 0.054
PG ₅ +PC _n	<u>0.888</u> ± 0.024	0.746 ± 0.033	0.731 ± 0.012	0.686 ± 0.066

indicator function. For assigning a higher ability for a student with less corrections, we multiply the value with -1 . **(b) Mean number of corrected parts (PC_n[#]):** this is a naïve version of PC_n and considers the mean number of the corrected parts of each student as the ability. Specifically, the mean number of the corrected parts of the student u is defined as $-\sum_{y_{uv}^{(n)} \in \mathcal{Y}_u^{(n)}} y_{uv}^{(n)} / |\mathcal{Y}_u^{(n)}|$, where $\mathcal{Y}_u^{(n)}$ is the set of correction outcomes for the student u . For assigning a higher ability for a student with less corrected parts, we multiply with -1 . **(c) Mean grades (PG[#]):** this is a naïve version of PG₁ and considers the mean assigned grades of each student as the ability. The mean grade is defined as $\sum_{z_{uv} \in \mathcal{Z}_u} z_{uv} / |\mathcal{Z}_u|$, where \mathcal{Z}_u is the set of grades assigned to the student u . **(d) PG₁, PG₃, PG₄, and PG₅:** existing peer assessment models.

7.2 Experimental setup

We implemented the models using the No-U-Turn Sampler (NUTS) [3], which is a variation of the Hamiltonian Monte Carlo. We executed four chains and they produce 5,000 samples in total. The initial 500 samples were ignored and the average of the rest samples were used as the estimated parameters.

We randomly generated 150 sets of candidate hyperparameters for each method. A method with a set of candidate hyperparameters produces the estimated student abilities. Their performance was evaluated using the groundtruth of 20% of the students. We then decided the best set of hyperparameters for the method and the final result for each method was evaluated by the remaining students. We performed this procedure five times and calculated the average.

Each method outputs the estimated ability of each student. We use the expertise levels assessed by the experts as the ground truth, and investigate how accurately each method

classifies the students with high expertise and those with low expertise. We specifically use the area under the ROC curve (AUC) as an evaluation metric.

7.3 Results

Table 2 shows the AUC scores of each method on different classification boundaries. Our peer correction models (PC_b and PC_n) demonstrate better or comparative performance to the existing peer grading models in detecting the students at the highest level; this supports the effectiveness of the peer correction results for estimating student abilities. We see that the “no-correction” cases only occur for high-ability students and the correction information is helpful for distinguishing between the “perfect students” and “almost perfect students”, both are likely to obtain the highest grades from the reviewers and the correction outcomes are required to classify them.

In contrast, the performance of peer correction models becomes inferior for detecting the students at lower levels, and PG[#] achieves the best performance for detecting the students at the lowest level; the average of the obtained grades is sufficiently informative for detecting low-ability students. Our methods would be beneficial for a situation where teachers aim to detect students who require advanced course materials or assignments.

The combined models of PG_{1,3,4,5}+PC_b outperform the corresponding PG_{1,3,4,5} in most cases; the outcomes of peer correction are useful for improving the student ability estimation. It is noteworthy that PG₅+PC_b achieves an AUC of 0.914 for classifying the students at A+ and the others. This result is brought by the capability of the combined models for capturing the inconsistencies between the outcomes of assessments and those of corrections.

The number of corrected parts can be more informative than simply considering whether a submission is corrected; in fact, PC_n is better than PC_b and PC_n^\sharp performs better than PC_b^\sharp ; however, $PG_{\{1,3,4,5\}} + PC_b$ outperforms $PG_{\{1,3,4,5\}} + PC_n$ in our experiments. Because there are more model variations in PC_n than PC_b , a more meticulous modeling for combining the PG models and the PC_n model would be required.

8. RELATED WORK

Peer assessment models are categorized into two groups: models for cardinal peer assessment and models for ordinal peer assessment. The former models target a situation where the outcomes are assigned in explicit numerical scores, such as five-point scores. In addition to the probabilistic models reviewed in Section 5, Walsh proposed PeerRank [13], an extension of PageRank for peer assessment. In ordinal peer assessment, each grader is shown multiple submissions and instructed to rank them. The Bradley–Terry model [2] has been applied for ordinal peer assessment [11, 8] and Mi et al. proposed to use the cardinal peer assessment models for ordinal peer assessment [6]. Although several probabilistic models for peer assessment have been studied, peer correction has not yet been investigated.

The design of peer assessment frameworks has been attempted to improve the reliability of evaluation. Kulkarni et al. ([4]) reported that the feedback about the grading bias to graders was beneficial for improving the reliability. Another work proposed to design peer assessment as a multiple choice task where a grader is instructed to choose the best submission [5]. Peer assessment mechanisms based on game theory have been introduced to derive accurate evaluations from peers [14].

Our peer correction models are very related to the models studied in the item response theory, which are for quantifying student abilities and item characteristics in educational tests. One of the simple item response theory model is the Rasch model [9] and our PC_b (given in Eq.(1)) model has a similar formulation to the Rasch model.

Besides peer assessment, probabilistic models for estimating grader reliability have been studied in crowdsourcing as well. Specifically, a two-stage framework was proposed where crowdsourcing workers in the first stage produce outputs, such as translations or logo designs, and another set of workers in the second stage evaluates the outputs [1]. Probabilistic models for estimating the reliability of each grader and the quality of each output in this two-stage framework have been proposed [1, 12]. Unlike peer assessment, the overlap between students (i.e., creators of outputs) and graders is not assumed in crowdsourcing.

9. CONCLUSIONS

We presented probabilistic models for peer correction, which are used for estimating the student abilities. We proposed two models: one considering whether a grader has corrected a submission, and the other utilizing the number of corrected parts in each submission. We also combined the peer correction models with the peer assessment models; this combination allows us to estimate the reliability of graders from the outcomes of peer corrections and those of peer assess-

ment by considering the consistency between the corrections and assessments. The experiments using the actual dataset of peer correction showed that the combination of peer correction models and peer assessment models was particularly effective in detecting high ability students.

In our models, we did not consider the importance of each corrected part; however, the importance levels differ among corrected parts in which minor corrections (e.g., adding a punctuation mark) and major corrections (e.g., paraphrasing) exist. A major correction would indicate the low quality of a submission and considering such factors is a promising direction to improve the ability estimation accuracy.

10. ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Number 15H01704 and 18K18105.

11. REFERENCES

- [1] Y. Baba and H. Kashima. Statistical quality estimation for general crowdsourcing tasks. In *ACM SIGKDD*, pages 554–562, 2013.
- [2] R. A. Bradley and M. Terry. The rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3):324–345, 1952.
- [3] M. D. Hoffman and A. Gelman. The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.
- [4] C. Kulkarni, K. P. Wei, H. Le, D. Chia, K. Papadopoulos, J. Cheng, D. Koller, and S. R. Klemmer. Peer and self assessment in massive online classes. *ACM Trans. Comput.-Hum. Interact.*, 20(6):33:1–33:31, 2013.
- [5] I. Labutov and C. Studer. JAG: a crowdsourcing framework for joint assessment and peer grading. In *AAAI*, pages 1010–1016, 2017.
- [6] F. Mi and D.-Y. Yeung. Probabilistic graphical models for boosting cardinal and ordinal peer grading in MOOCs. In *AAAI*, pages 454–460, 2015.
- [7] C. Piech, J. Huang, Z. Chen, C. Do, A. Ng, and D. Koller. Tuned models of peer assessment in MOOCs. In *EDM*, pages 153–160, 2013.
- [8] K. Raman and T. Joachims. Methods for ordinal peer grading. In *ACM SIGKDD*, pages 1037–1046, 2014.
- [9] G. Rasch. *Probabilistic Models for Some Intelligence and Attainment Tests*. 1960.
- [10] J. W. Ratcliff and D. E. Metzener. Pattern matching: the Gestalt approach. *DDJ*, 13(7):46, 1988.
- [11] N. B. Shah, J. K. Bradley, A. Parekh, M. Wainwright, and K. Ramchandran. A case for ordinal peer-evaluation in MOOCs. In *NIPS-DDE*, 2013.
- [12] T. Sunahase, Y. Baba, and H. Kashima. Pairwise HITS: quality estimation from pairwise comparisons in creator-evaluator crowdsourcing process. In *AAAI*, pages 977–984, 2017.
- [13] T. Walsh. The PeerRank method for peer assessment. In *ECAI*, pages 909–914, 2014.
- [14] W. Wu, C. Daskalakis, N. Kaashoek, C. Tzamos, and M. Weinberg. Game theory based peer grading mechanisms for MOOCs. In *L@S*, pages 281–286, 2015.

Implicit and Explicit Emotions in MOOCs

Munira Syed
University of Notre Dame
msyed2@nd.edu

Malolan Chetlur
IBM Research, India
mchetlur@in.ibm.com

Shazia Afzal
IBM Research, India
shaafzal@in.ibm.com

G. Alex Ambrose
University of Notre Dame
gambrose@nd.edu

Nitesh V. Chawla
University of Notre Dame
nchawla@nd.edu

ABSTRACT

Understanding the affect expressed by learners is essential for enriching the learning experience in Massive Open Online Courses (MOOCs). However, online learning environments, especially MOOCs, pose several challenges in understanding the different types of affect experienced by a learner. In this paper, we define two categories of emotions, *explicit* emotions as those collected directly from the student through self-reported surveys, and *implicit* emotions as those inferred unobtrusively during the learning process. We also introduce positivity as a measure to study the valence reported by students chronologically, and use it to derive insights into their emotion patterns and their association with learning outcomes. We show that implicit and explicit emotions expressed by students within the context of a MOOC are independent of each other, however, they correlate better with students' behavior compared to their valence.

Keywords

MOOC, emotions, discussion forum, valence, surveys

1. INTRODUCTION

The exploration of emotions expressed by students in Massive Open Online Courses (MOOCs) has caught the attention of researchers for improving the remote and non-contact learning experience [28, 8, 16, 5]. A few examples of these studies infer emotions of students from their behavior [16], surveys collected during the course [8, 1], clickstream data and discussion forums [28, 5]. The relationship between students' emotions and their behavior, learning outcomes, engagement, and dropout within the MOOC context is established in [1, 25, 21].

Emotions experienced by students during a course impact their behavior and learning outcomes [15, 19]. Detecting the emotion experienced during learning is difficult, and various methods have been employed for this purpose. The methods used to sample emotions mainly fall into three categories

as outlined by [27]. The first category consists of methods that take snapshots of students' emotions during the course through survey questionnaires. These methods are intrusive to the learning process and are usually self-reported and subjective in nature. The second category detects emotions during the learning process and includes methods that sample emotions non-intrusively like facial expression detection, conversations, gaze detection, and analysis of text data generated by student interactions within the course [9, 10]. The third category measures emotions after the learning process. The first two categories are relevant to our paper. In [27], the methods in the second category are assumed to counteract the limitations of the methods in the first category. Therefore, in our study we use two categories of emotions to get a more complete view of students' emotional states. In this paper, we measure explicit emotions as the emotions recorded from student's self-reported surveys and Self-Assessment Manikins (SAMs), and implicit emotions as those from the open discussion forum posts of students.

Emotions measured in association with learning seem to be short-lived and last for a few seconds to minutes [15]. Since the emotions were expressed by students in this MOOC at different, non-uniform points in time, one of the challenges of analyzing such a series is the spontaneity of emotions. As the emotions are surveyed after the end of a video or module, we only get a snapshot of the students' emotions during the course [27]. Between two consecutive surveys, a student's emotions can not only change multiple times, but also be conflicting, as students can experience multiple emotions simultaneously [1], which could hinder a chronological analysis of the emotions. However, even if students' emotions are spontaneous and likely to be fraught with missing data, there might be a trend to their emotions over time. An approach that leverages this idea has been proposed in [7], where the positive affect experienced by an individual is averaged over a period of time while the negative reports are ignored. Inspired by this technique, we also calculate the "positivity" of students at each point of the reported emotions and derive a positivity sequence instead of an emotion sequence. This positivity sequence is expected to be more stable over time as compared to the emotion sequence.

We study the implicit and explicit emotions expressed by the MOOC students through the following research questions.

RQ1: Are the explicit and implicit emotions expressed within a MOOC context similar? Can one be used as a proxy for

Munira Syed, Malolan Chetlur, Shazia Afzal, G. Alex Ambrose and Nitesh V. Chawla "Implicit and Explicit Emotions in MOOCs" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 432 - 437

the other or are both of them equally important for characterizing a student's emotional state?

RQ2: What do the combined (explicit plus implicit) emotional states and positivity sequences characterize about a student's learning?

To the best of our knowledge, this is the first attempt at investigating the effect of explicit and implicit emotion categories within a MOOC context. We find that implicit and explicit emotions expressed by students are indeed different and both are necessary to characterize student emotions. We also see that combined positivity values correlate relatively well with behavior compared to their valence values.

2. RELATED WORK

The comparison of self-reported metrics like emotions and performance in self-regulated learning and other educational contexts has been studied and generally found to be inconsistent with the measured reports [14, 26, 29]. While many of these studies measure the alignment of students' achievement calibration with their actual performance [13, 26, 14], we aim to compare the self-reported emotions of students in MOOCs against the emotions we measure from their behavior in the MOOC, in the form of interactions on the discussion forum. A direct comparison of these methods with ours is infeasible because of the difference in instrumentation and methodology. However, we will compare our general observations with the trends in literature.

We use students' self-reports of emotions along with Self-Assessment Manikin (SAM) as the explicit measures of students' emotions. Self-reports are a very common way of measuring students' emotions because of their subjective nature [11]. Collecting students' emotions through surveys is easy to deploy on a large-scale and is low cost [11], which makes them favourable for use in MOOCs [1]. SAM is a non-verbal assessment technique that allows people to rate their pleasure, represented as valence in our case, on an ordinal scale [4]. SAMs have been used to measure emotion in online learning environments [6, 8].

Among the techniques available for detecting the implicitly expressed emotions of students, analyzing emotions from texts is one of the least invasive ways of detecting students' emotions [17, 22]. Using discussion forums to detect students' emotions in MOOCs is becoming prominent due to its unobtrusiveness and low instrumentation [28]. Many sentiment analysis techniques for detecting valence from text including the word-affect lexicon used in this paper are listed in [18], and education has been noted as one of the applications of sentiment analysis. We use Warriner's [24] word-affect lexicon to calculate the valence values of words in the discussion forum records. The effectiveness of Warriner's word-affect lexicon [24] for sentiment analysis has been demonstrated for detecting sarcasm [20], finding geographical locations associated with happier tweets [12], etc. This automatic method to detect affect from discussion forum data enables a scalable way to glean implicit affect in MOOCs from a large number of forum posts. Sentiment analysis polarity techniques were applied on discussion forum posts in [25]. In [28], a Mechanical Turk is used to obtain confusion ratings among students through simple features like counting the number of question marks to predict

**Table 1: 1. Number of students vs. SAM surveys
2. Number of students vs. SAM scores**

SAM survey	No. of students	SAM score	No. of students
1	4111	1	3204
2	2815	2	4355
3	1354	3	1557
4	906	4	295
5	326	5	101

the level of confusion in the discussion forum posts. They also use Linguistic Inquiry and Word Count (LIWC) to consider negation words and phrases as an indicator of potential confusion, and clickstream patterns (eg. quiz-quiz-forum) as a feature for detecting confusion. Previous research on using the discussion forum to estimate student retention and performance is complicated due to a vast amount of missing and imbalanced data [3]. We also face challenges to detect implicit emotions in the midst of context-specific terms.

3. DATA DESCRIPTION

3.1 Course Description

We use the data from the introductory course on Statistics called "I Heart Stats" for our study. This was a self-paced MOOC on the EdX platform, and the entire course content was released at the start of the course. The course had nine modules, with the ninth module being for the assessment of the overall course. During the course, students were asked to self-report their emotions and valence through emotion surveys and SAM surveys respectively. Initially 24,279 students were enrolled in the course, however, only less than 15,000 students had activity in the first two weeks. Finally, only 1,941 students completed it. Of all the students, 1,629 responded to at least one emotion or SAM survey, and participated in the discussion forum as well. Only these students have been included in the Analysis section of the paper as these are the only students generating both implicit as well as explicit emotions. Note that students completing the course are likely to have longer sequence lengths. Students not interacting with the discussion forum but are still part of the course cannot be included in the analysis leading to an overrepresentation of active users.

3.2 Explicit Emotions

Emotion Surveys: Of all the students, 6,100 submitted 21,448 emotion surveys. During the course, 12 emotion surveys were conducted in which students self-reported their current emotional state. This was optional and students could choose multiple of a list of 15 emotions: *anger, anxiety, boredom, confusion, contentment, disappointment, enjoyment, frustration, hope, hopelessness, isolation, pride, relief, sadness, and shame*. Further details can be found in [1]. The valence values of these emotions were calculated using Warriner's lexicon [24], with a scale of 1 to 9 and 5 being neutral. We shift the scale to $[-4, 4]$ to bring the neutral valence to 0. In the case of multiple emotions being expressed, the associated valence values were averaged to obtain one valence value per survey. Thus, the surveys have positive $(0, 4]$, negative $[-4, 0)$, and neutral $\{0\}$ valence values.

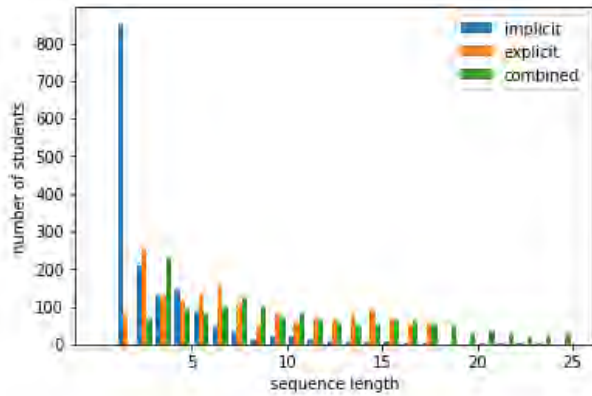


Figure 1: Histogram of implicit, explicit, and combined sequence lengths (until sequence length of 25)

SAM Surveys: A total of 5 SAM surveys, using a 5-point scale, were conducted in this MOOC. The SAM score represented in Table 1 ranges from 1 to 5 with 1 being the least and 5 being the highest state of pleasure. As the distribution of the number of students corresponding to each SAM score is normal, we convert this scale to an interval scale in the range $[-4, 4]$ linearly. In total, 5,363 students have submitted 9,512 SAM surveys with the rest of the details shows in Table 1.

3.3 Implicit Emotions

The discussion forum is a platform that students use to interact with each other, the instructor, and teaching assistant of the MOOC. In total, 1,717 students generated 5,322 discussion forum records. The posts, comments, and replies (i.e. records) on the discussion forum are used to infer the implicit emotions of students.

We use Warriner’s word-affect lexicon [24] to calculate the valence values of discussion form records. The tokenized words in tweets are used to calculate the mean valence value of the tweet using Warriner’s word-affect lexicon. We use a similar approach to calculate valence values for discussion forum records using the following steps: (i) Tokenize the records to get a list of words, (ii) Remove the stop words from the list, (iii) Make a list v of valence values associated with a word using the lexicon, if present, after re-scaling them between $[-4, 4]$, (iv) Multiply the valence values of words/phrases that follow a negative word with -1 (eg. not, never), and (iv) Return the average valence value of list v .

3.4 Combined Emotions

Throughout the course, students have multiple opportunities, explicit or implicit, to express their emotions. The 12 emotion surveys, 5 SAM surveys, and valence values calculated from discussion forum records were interleaved and ordered chronologically for each student to form a combined sequence of valence values.

A histogram of the number of reports corresponding to the number of students in Figure 3.4 shows that the highest number of students (14%) has a maximum combined sequence length of 3 with the number of students tapering

down after that point. The maximum number of reports corresponding to a student is 74, as this student was very active in the discussion forum.

To mitigate the spontaneous nature of emotions, we calculate the positivity of students at each report from the valence sequence values. Thus, if a student reports one negative emotion among a string of positive emotions, the impact of the negative emotion is reduced because of the previously expressed positive emotions. We define positivity as follows.

Positivity: Let r_1, r_2, \dots, r_n be the reports made by a student until element n such that: $timestamp(r_{i-1}) < timestamp(r_i)$ for all i . The valences are normalized between $[-1, 1]$, instead of $[-4, 4]$, by dividing them by 4. Let p_1, p_2, \dots, p_m be the positive normalized valences where $m \leq n$ and $m + 1 > n$. The positivity at the n th element is given by $(p_1 + p_2 + \dots + p_m)/n$.

In other words, an element of the positivity sequence is calculated by averaging over only the positive valences in the sequence until that element. Since students have reported more positive than negative valences both explicitly and implicitly, calculating negativity instead of positivity would lead to extremely sparse sequences.

4. ANALYSIS

4.1 Calculated Valences

Section 3.3 lists the steps to calculate the valence values of the discussion forum records. To validate these valence values, 440 samples of the discussion forum records were manually annotated by three human raters in which each rater chooses one, two, or none of the 15 emotion choices that students had for their emotion surveys. The fourth rater is the calculated valence. We use Fleiss’ Kappa [2] to calculate the inter-rater agreement by converting the valence scores to positive, negative, or zero valence. The inter-rater agreement of the three human raters is 0.457 (moderate agreement), whereas the inter-rater agreement of the four raters including the calculated valences is 0.218 (fair agreement) [23]. While the agreement including the calculated valences is lower, it is adequate, and so we use the calculated valence of these discussion forum records as the implicit valence values.

4.2 Implicit vs. Explicit features (RQ1)

Both implicit and explicit sequences are instances of irregular time-series data. However, since emotion data is spontaneous and might change multiple times between consecutive reports [15], averaging, downsampling, interpolating or duplicating valence values in an emotion sequence might misrepresent the true emotional trajectory of the student.

4.2.1 Feature vectors description

Since the valence sequences are not uniform in length, we create fixed length feature vectors for analysis. The features are used in Sections 4.2.2 and 4.2.3 with their description given: (i) *pos*: ratio of the number of positive valences to the total length of the sequence (ii) *neg*: ratio of the number of negative valences to the total length of the sequence (iii) *neu*: ratio of the number of neutral valences to the total length of the sequence (iv) *trans*: ratio of the number

Table 2: Corr. between implicit & explicit features

Features	Pearson's r	Spearman's rho
pos	0.0401	0.0696**
neg	0.0413*	0.102***
neu	0.0150	0.0380
seq_len	0.346***	0.422***
trans	0.125***	0.162***
neg_pos	0.113***	0.165***
pos_neg	0.0805**	0.127***
range	0.243***	0.257***

*:p-val.<0.1, **:p-val.<0.05, ***:p-val.<0.0001

of transition of valences from positive to negative or vice versa in the sequence to the sequence length (v) *pos_neg*: ratio of the number of transition of valences from positive to negative to the sequence length (vi) *neg_pos*: ratio of the number of transition of valences from negative to positive to the sequence length (vii) *range*: calculated by subtracting the minimum valence value from the maximum valence value expressed (To normalize the value the resulting range is divided by 8, as the valence values lie in the range [-4, 4].) (viii) *seq_len*: length of the valence sequence (integral value).

4.2.2 Correlation

In Table 2, we see that *pos*, *neg*, and *neu*, as defined in Section 4.2.1, between implicit and explicit emotions of students are not correlated with each other. This shows that both types of sequences are somewhat independent of each other and might show different insights into students' affect. There are relatively few neutral discussion forum records which is why its correlation with completion is not significant. That is why transitions from neutral to positive and negative valences, and vice-versa have been left out of the features list. The sequence lengths seem to be mildly correlated showing that students reporting more emotions in the emotion surveys were also more likely to submit more records in the discussion forum. This correlation is expected since the number of students with larger sequence lengths decreases as seen from Figure 3.4.

4.2.3 Clustering of Feature Vectors

We cluster the 7-dimensional feature vector to identify groups of similar students using K-Means. To visualize the clusters created, we decompose the 7-dimensional feature vectors of students' implicit and explicit emotion sequences to a 2-dimensional space using Principal Component Analysis (PCA) separately. The PCA decomposition in Figure 4.2.3 shows very separable clusters in the 2-dimensional space. The explicit clusters have significantly different ratios of course completion: orange: 37.2%, purple: 25.5%, olive: 51.9%. Similarly, the completion ratios of the implicit clusters are: red: 34.5%, blue 32.6%, green: 60.3%, with the green cluster having significantly more students completing the course than the other two.

4.3 Combined sequence features (RQ2)

From the previous subsection, we saw that implicit and explicit sequences are not identical and should both be incorporated into a student's valence trajectory. So we use both

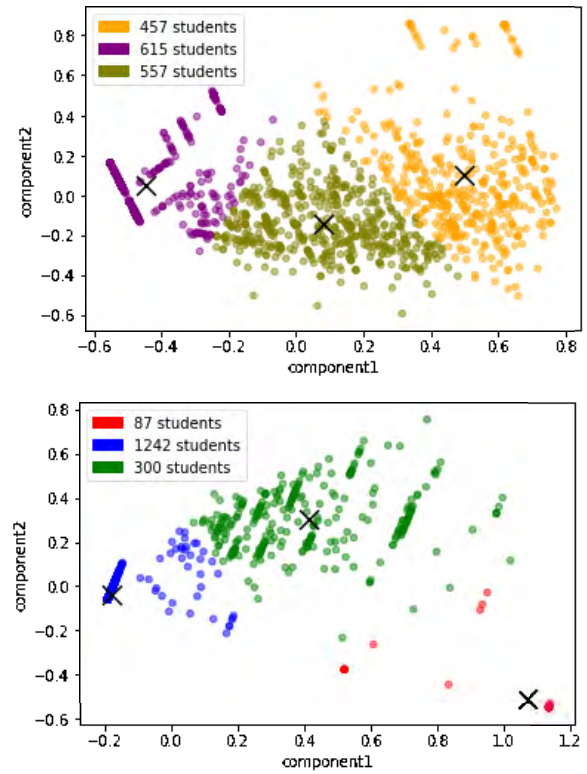


Figure 2: PCA decomposition of explicit (top) and implicit (bottom) seq. clusters ('x': cluster centers)

implicit and explicit sources of emotions ordered by time to generate a combined valence sequence for students. The features from Section 4.2.1 are used in the analysis below.

4.3.1 Correlation of features with completion

We generate the 7-dimensional feature vector from the combined valence sequence for each as defined in Section 4.2.1 and show the correlation of each dimension with completion in Table 3. Completion is defined by a student reaching module 8 [1]. We see that *seq_len* has the highest correlation with completion possibly because sequence length could act as proxy for the amount of time students spent in the course. A similar reasoning might hold for *trans*. The *pos*, *neg*, or *neu* features do not seem to be correlated with completion. However, *neg_pos* seems to be better correlated with

Table 3: Corr. of combined vectors with completion

Feature	Pearson's r	Spearman's rho
pos	-0.0549***	-0.110***
neg	0.0807***	0.156***
neu	-0.0382**	0.0828***
neg_pos	0.215***	0.300***
pos_neg	0.112***	0.201***
trans	0.186***	0.223***
seq_len	0.523***	0.460***
range	0.390***	0.392***

: p-val. <0.05, *: p-val. <0.0001

Table 4: Corr. of features with quiz performance

Features	average	minimum	maximum
range	-0.0804*	-0.181***	0.0735*
seq_len	-0.232***	0.0681*	-0.405***

*: p-val.<0.1, **: p-val.<0.05, ***: p-val.<0.0001

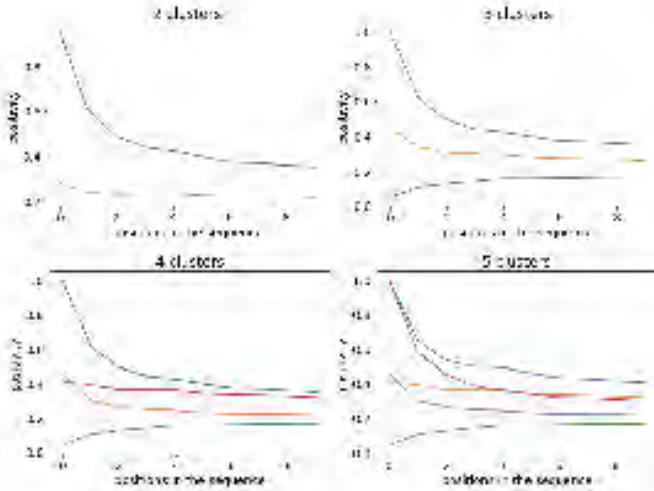


Figure 3: Positivity Clustering of Combined Seqs.

completion than *pos_neg*. This supports our intuition that students transitioning from a negative to positive emotional state are more likely to stay in the course, compared to the other way round. The feature *range* is better correlated with completion than *trans* which indicates that higher intensity of changes in emotions is more likely to result in completion.

4.3.2 Correlation of features with Quiz Performance

The performance score of students for a quiz is normalized between 0 and 1. The average, minimum, and maximum performance score of the quizzes (total 4) that students have attempted is used as the y-variable for correlation. The features that are significantly correlated with these statistics using Pearson’s correlation are in Table 4. While the negative correlation with *seq_len* is unsurprising given that harder quizzes are towards the end of the course, the positive correlation with *range* suggests that student who experience extreme emotions tend to perform better.

4.4 Positivity clustering (RQ2)

We compare fixed length positivity sequences by clustering the first 10 elements of 767 students who have a sequence length of at least 10. We see that $k=3$ is the highest number that shows no overlap of cluster centers. While there is no significant difference between the clusters for quiz performance, the difference between clusters in terms of quiz participation using ANOVA is significant at $p\text{-value} < 0.05$. Specifically, in the $k=3$ chart in Figure 4.4, there are more students in the most positive (green) cluster that do not submit a single quiz (29.3%) than the other two clusters (20%). A possible explanation is that students had trouble with the quizzes and the ones who did not attempt them were more likely to be happier. All three cluster centers converge to-

wards a narrow range of positivity, suggesting that students tend towards the same positivity in the course even though they started out differently.

5. DISCUSSION AND FUTURE WORK

Similar to the studies [14, 26, 29], we found that the self-reported emotions did not reflect the implicitly measured emotions. Clustering students by their emotion sequence had different ratios of students that completed the course in each cluster. This observation is similar to what [14] found about different learning strategies and activity of students. To investigate whether the temporally proximal self-report was correlated with the outcome completion, we measured the correlation of the last reported valence and the final positivity in the students’ sequences with completion. However, similar to [29], we found no correlation. This suggests that the proximity of students’ emotions to the outcome completion does not have a bearing on completion.

Through RQ1, we show that both the implicit and explicit emotion sequences are independent of each other and contribute different emotional information. Through RQ2, we showed that students tend to converge towards the same positivity even though they start out differently, indicating that they end up feeling the same way. This might be because of external factors that remained constant for all the students, e.g., how the course was conducted, possibly explaining the lack of correlation with the course outcomes. We see significant differences between these clusters in quiz participation but not in other learning outcomes. This may be because students who did not attempt the quizzes did not struggle through the course and remained relatively happy. Our results show that there is potential for identifying different groups of students that participate in a MOOC. Table 2 shows that the explicit and implicit sequences are associated with behavior, but not valence. One of the possible reasons is that students who participate more in the discussion forum tend to submit more surveys as well but the two types of sequences do not corroborate each other in valence. From Table 3, we also observe that students who feel negatively about the course and then transition to a positive emotional state are more likely to stay in the course. We found that the range of valence that students experience is more indicative of their course completion and quiz performance possibly because the students who struggle through the course report higher valence values after achieving their course objectives, resulting in their highly varied emotions.

A limitation of our work is our sentiment analysis technique that uses a bag-of-words model with the discussion forum records only and does not consider other implicit measures of emotions. In this work, we have only relied on a single word-affect lexicon. However, we can make the calculated valence values more stable by triangulating the valences with other lexicons. We would also like to improve granularity and quantify the extra information conveyed by either type of emotion sequence. Even so, as most emotion research in MOOC relies on only one category of emotions, we conclude that it might be advantageous for researchers in this area to supplement their current method with a method from the other category of emotions. It is important to continue exploring emotions in MOOCs in pursuit of goals such as the personalization of MOOCs, improving the emotional well-

being of students, and the design of MOOCs.

6. ACKNOWLEDGEMENTS

We thank John Dillon for his contributions to this project. This work was supported in part by the National Science Foundation (NSF) Grant IIS-1447795.

7. REFERENCES

- [1] S. Afzal, B. Sengupta, M. Syed, et al. The ABC of MOOCs: Affect and its inter-play with behavior and cognition. In *ACII*, pages 279–284. IEEE, 2017.
- [2] M. Banerjee, M. Capozzoli, L. McSweeney, et al. Beyond kappa: A review of interrater agreement measures. *Canadian journal of statistics*, 27(1):3–23, 1999.
- [3] Y. Bergner, D. Kerr, and D. E. Pritchard. Methodological challenges in the analysis of MOOC data for exploring the relationship between discussion forum views and learning outcomes. *International Educational Data Mining Society*, 2015.
- [4] M. M. Bradley and P. J. Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994.
- [5] J. C. Cheng. An exploratory study of emotional affordance of a massive open online course. *European Journal of Open, Distance and e-learning*, 17(1):43–55, 2014.
- [6] K.-H. Cheng, H.-T. Hou, and S.-Y. Wu. Exploring students’ emotional responses and participation in an online peer assessment activity: A case study. *Interactive Learning Environments*, 22(3):271–287, 2014.
- [7] E. Diener, R. J. Larsen, S. Levine, et al. Intensity and frequency: dimensions underlying positive and negative affect. *Journal of personality and social psychology*, 48(5):1253, 1985.
- [8] J. Dillon, N. Bosch, M. Chetlur, et al. Student emotion, co-occurrence, and dropout in a MOOC context. In *EDM*, pages 353–357, 2016.
- [9] S. D’Mello, A. Olney, C. Williams, et al. Gaze tutor: A gaze-reactive intelligent tutoring system. *International Journal of human-computer studies*, 70(5):377–398, 2012.
- [10] S. K. D’mello, S. D. Craig, A. Witherspoon, et al. Automatic detection of learner’s affect from conversational cues. *User modeling and user-adapted interaction*, 18(1-2):45–80, 2008.
- [11] J. A. Fredricks and W. McColskey. The measurement of student engagement: A comparative analysis of various methods and student self-report instruments. In *Handbook of research on student engagement*, pages 763–782. Springer, 2012.
- [12] L. Gallegos, K. Lerman, A. Huang, et al. Geography of emotion: Where in a city are people happier? In *WWW*, pages 569–574. ACM, 2016.
- [13] L. S. Garavalia and M. E. Gredler. An exploratory study of academic goal setting, achievement calibration and self-regulated learning. *Journal of instructional psychology*, 29(4):221–231, 2002.
- [14] D. Gašević, J. Jovanovic, A. Pardo, et al. Detecting learning strategies with analytics: Links with self-reported measures and academic performance. *Journal of Learning Analytics*, 4(2):113–128, 2017.
- [15] A. Graesser and S. K. D’Mello. Theoretical perspectives on affect and deep learning. In *New perspectives on affect and learning technologies*, pages 11–21. Springer, 2011.
- [16] D. Leony, P. J. M. Merino, J. A. R. Valiente, et al. Detection and evaluation of emotions in massive open online courses. *J. UCS*, 21(5):638–655, 2015.
- [17] Z. Liu, W. Zhang, J. Sun, et al. Emotion and associated topic detection for course comments in a mooc platform. In *2016 International Conference on Educational Innovation through Technology (EITT)*, pages 15–19. IEEE, 2016.
- [18] S. M. Mohammad. Sentiment analysis: Detecting valence, emotions, and other affectual states from text. In *Emotion measurement*, pages 201–237. Elsevier, 2016.
- [19] R. Pekrun, T. Goetz, W. Titz, et al. Academic emotions in students’ self-regulated learning and achievement: A program of qualitative and quantitative research. *Educational psychologist*, 37(2):91–105, 2002.
- [20] A. Rajadesingan, R. Zafarani, and H. Liu. Sarcasm detection on twitter: A behavioral modeling approach. In *WSDM*, pages 97–106. ACM, 2015.
- [21] A. Ramesh, D. Goldwasser, B. Huang, et al. Understanding mooc discussion forums using seeded LDA. In *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 28–33, 2014.
- [22] P. Rodriguez, A. Ortigosa, and R. M. Carro. Extracting emotions from texts in e-learning environments. In *2012 Sixth International Conference on Complex, Intelligent, and Software Intensive Systems*, pages 887–892. IEEE, 2012.
- [23] A. J. Viera and J. M. Garrett. Understanding interobserver agreement: the kappa statistic. *Fam med*, 37(5):360–363, 2005.
- [24] A. B. Warriner, V. Kuperman, and M. Brysbaert. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior research methods*, 45(4):1191–1207, 2013.
- [25] M. Wen, D. Yang, and C. Rose. Sentiment analysis in MOOC discussion forums: What does it tell us? In *EDM*. Citeseer, 2014.
- [26] P. H. Winne and D. Jamieson-Noel. Exploring students’ calibration of self reports about study tactics and achievement. *Contemporary Educational Psychology*, 27(4):551–572, 2002.
- [27] M. Wosnitza and S. Volet. Origin, direction and impact of emotions in social online learning. *Learning and instruction*, 15(5):449–464, 2005.
- [28] D. Yang, M. Wen, I. Howley, et al. Exploring the effect of confusion in discussion forums of massive open online courses. In *L@S*, pages 121–130. ACM, 2015.
- [29] M. Zhou and P. H. Winne. Modeling academic achievement by self-reported versus traced goal orientation. *Learning and Instruction*, 22(6):413–419, 2012.

Planning Gamification Strategies based on User Characteristics and DM: A Gender-based Case Study

Armando M. Toda
Durham University, United Kingdom
University of Sao Paulo, Brazil
armando.toda@usp.br

Wilk Oliveira
University of Sao Paulo
Institute of Mathematics and Computer Science
Sao Carlos, Brazil
wilk.oliveira@usp.br

Lei Shi
University of Liverpool
Liverpool, United Kingdom
lei.shi@liverpool.ac.uk

Ig Ibert Bittencourt
Federal University of Alagoas
Maceio - Alagoas
ig.ibert@gmail.com

Seiji Isotani
University of Sao Paulo
Institute of Mathematics and Computer Science
Sao Carlos - Brazil
sisotani@icmc.usp.br

Alexandra Cristea
Durham University
Department of Computer Science
Durham, United Kingdom
alexandra.i.cristea@durham.ac.uk

ABSTRACT

Gamification frameworks can aid in gamification planning for education. Most frameworks, however, do not provide ways to select, relate or recommend how to use game elements, to gamify a certain educational task. Instead, most provide a “one-size-fits-all” approach covering all learners, without considering different user characteristics, such as gender. Therefore, this work aims to adopt a data-driven approach to provide a set of game element recommendations, based on user preferences, that could be used by teachers and instructors to gamify learning activities. We analysed data from a novel survey of 733 people (male=569 and female=164), collecting information about user preferences regarding game elements. Our results suggest that the most important rules were based on four (out of nineteen) types of game elements: Objectives, Levels, Progress and Choice. From the perspective of user gender, for the female sample, the most interesting rule associated Objectives with Progress, Badges and Information (confidence=0.97), whilst the most interesting rule for the male sample associated also Objectives with Progress, Renovation and Choice (confidence=0.94). These rules and our descriptive analysis provides recommendations on how game elements can be used in educational scenarios.

1. INTRODUCTION

Gamification has been a widely popular phenomenon in the past few years, being used in various domains, including that of education. Gamification is defined as the use of game elements outside their scope (*i.e.*, games or game playing) [1, 2]. However, educators often may not be familiar with

specific game-related concepts, or know how to use game elements, or may not have the resources or time necessary [3, 4, 5]. A solution is to employ conceptual gamification frameworks [6]. Still, existing frameworks lack resources and explanations on how to use game elements appropriately [5], especially when considering user preferences affected by demographic differences. Understanding users’ characteristics, such as gender, may be especially beneficial, *e.g.*, in STEM education, where the well-known problem of ‘the leaking STEM pipeline’¹ occurs [7].

In this paper, we apply a data-driven approach to provide insights into the educational domain, via the research question: “*How can gender differences in preferences about gamification elements be used to support gamification design?*” We conducted a very large survey (808 raw answers) allowing respondents to rank gamification elements. We based these elements on the works of Dignan [8] and Toda *et al.* [5], due to (a) the relatively large number and variety of elements, (b) the availability of synonyms used. Next, we used an unsupervised learning algorithm to generate Association Rules to find patterns within the dataset, in order to understand relations among these elements, based on the users’ genders. Our main contributions are: (a) a survey² for extracting preferences for gamification for education, applied to a large, varied number of respondents; (b) extracting gamification elements relevant to the different genders, for the educational domain; (c) extracting relations between these elements, relevant to the different genders; (d) insights into users’ acceptance of specific game elements, or groups thereof (and their relations).

2. RELATED WORKS

As there are very few frameworks focusing on gamification in education domains, we discuss: (i) existing models related to game elements, (ii) gamification studies on user characteristics, (iii) planning of gamification.

¹dropout in STEM education

²<https://forms.gle/hFgTT7kCqBKLqiPd8>

Armando Maciel Toda, Wilk Oliveira, Lei Shi, Ig Ibert Bittencourt, Seiji Isotani and Alexandra Cristea "Planning Gamification Strategies based on User Characteristics and DM: A Gender-based Case Study" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 438 - 443

Yee and Marczewski both proposed models on how to use game elements, using also large data collections [9, 10]; however, their focus is different: they collected (a) players' motivations towards online RPGs and (b) generic gamified applications. [10] only provides recommendations of elements for behavioural profiles, but not user demographic characteristics, such as gender. Yee's model additionally analysed behaviours from a gender perspective, but only for online RPG (World of Warcraft) players exclusively. A recent study by Shi and Cristea [11, 12] proposed a model and a set of recommendations based on the Self-Determination Theory [13]. Their Motivational Gamification Strategies related game elements with each construct of the SDT, *i.e.*, Autonomy, Competence and Relatedness, and implemented them [11], achieving positive results for each construct. Such studies show how motivational theories and gamification constructs can be related, as well support gamification in education. They however do not support the design process of gamification for educators and teachers.

Denden *et al.* [14] conducted an experiment analysing user preferences ($N = 120$) over eight game elements within a gamified educational system, based on personality traits (the famous 'Big Five' [15]). According to the authors, only extraversion, openness and conscientiousness affected students' preferences for particular game elements. The authors also stated the importance of this kind of recommendation to designers and instructors when gamifying their learning environments. However, the gamification in education literature lacks studies which relate the acceptance and influence of game elements with users' genders, involving large-scale data [16]. One recent study [17] conducted an experimental study aiming at identifying differences between male and female users ($N = 70$) towards 'gaming the system' behaviours. It was shown that game elements led to male users decreasing their undesired behaviours; moreover, female users felt less competent than male users. Nevertheless, although the results are interesting, the number of students who were analysed is still relatively small, with students from within a course context - whereas our study has a wider scale and variety of participants.

Toda *et al.* [5] proposes a framework for blended classroom environments using social networks, via a list of recommendations (names of gamified strategies) based on previous studies in gamification in education. They also apply Dignan's game elements classification [8]. However, the gamified strategies proposed are solely based on literature. Nevertheless, their positive results show that game elements are suited for educational environments (*e.g.*, classroom and digital platforms). As noted, other gamification frameworks focused on specific domains (*e.g.* Computational Thinking [18]). Klock *et al* [19]'s framework is usable for adaptive system. Still, Mora *et al* [6] note that this framework focuses on the researchers, rather than the stakeholders (teachers and instructors) and presents limited recommendations on game elements usage.

Thus, whilst gamification shows potential benefits for educational applications, the gender differences in preference towards specific game elements needed further, large-scale, systematic studies, to better provide support for Data-Driven Gamification Design, as tackled by our current paper.

3. DATASET AND METHODS

Our survey on game elements contains 29 questions. The first part collected demographic information (age, favourite game setting, and gender). The second part asked to what extent certain game elements were relevant to users in the gamified educational system context, through a Likert Scale, from 1 "I think this element is irrelevant to me" to 5 "I think this element is highly relevant to me". The game elements used and their advantages and potential drawbacks are presented in <https://tinyurl.com/y44kqvn5> based on [8] and used in [5] in an educational domain.

Additionally to theoretical motivations, we further validated the selected gamification elements with 4 specialists in gamification, who were also teachers, via an interview, verifying the specialists' acceptance of the used elements, concepts, as well as questions cohesion. Finally, a pilot survey with 18 people verified the time spent and the consistency of the questions, before launching the main survey <https://goo.gl/forms/d0i5WosBcMVWvQAK2>. We then recruited surveyees through social networks, forums and digital environments used by people who play games.

In total, we collected 808 raw answers. Further cleaning removed data from users who: (a) did not answer all questions; (b) claimed not having played any digital games; (c) were of age < 0 or age > 90. Then, we analysed our population characteristics based on demographic data. As the normality test showed a non-normal distribution, a Mann-Whitney test [20] was used to compare males and females.

Finally, we used association rule mining to analyse the relations amongst our data, based on gender. Unsupervised learning was used as we do not have any predefined labels (outputs) and also to understand the relations between the elements (different from clustering which create groups based on all variables of the dataset). The algorithm analyses the items' frequency (support) and renders a level of confidence, ranging from 0 to 1 (where 1 is the maximum confidence). The confidence can also be supported by conviction [21], lift and leverage -- both measuring the independence of items.

4. RESULTS AND DISCUSSION

4.1 Gender differences

After filtering, we retrieved 733 valid answers (90.72%). We applied Cronbach's α on the second group of questions (as game elements were based on a Likert scale) and achieved an $\alpha = 0.83$ (high reliability factor [22]). Our sample is varied in terms of age (ranging from 13 to 68), but limited in terms of experience in playing (at least a year: by design and filtering) and country of origin (Brazil; due to convenience sampling). Nevertheless, the sampling size is much larger than the recommended one ($733 \gg 384$; people playing online games estimated at 700 mio; confidence level 95%).

We further organised our valid answers into two groups: males ($N = 569$) and females ($N = 164$), and verified the distribution of the data using the Shapiro-Wilk normality test. The result showed that our data rejected the null hypothesis ($p < 0.05$), so we adopted non-parametric tests in further analyses. Table 1 summarises the result.

Table 1: Relevance of game elements, averaged per gender

Element	Gender (mean)		Mann-Whitney	
	Female	Male	W	p-value
Point	3.76	3.68	44836	0.429
Level	4.14	4.21	48418	0.427
Cooperation	3.62	3.86	52306	0.013
Competition	3.26	3.56	53016	0.006
Renovation	4.16	3.78	35878	2.36e-03
Progress	4.24	4.32	48856	0.312
Objective	4.41	4.4	45902	0.791
Puzzles	4.14	3.91	40636	0.008
Novelty	4.05	4.16	49530	0.197
Chances	3.68	3.61	44901	0.447
Social Pressure	3.43	3.65	51142	0.05
Acknowledgement	3.85	3.73	44673	0.387
Data	4.05	4.09	46675	0.994
Scarcity	3.16	3.42	52468	0.011
Choice	4.07	4.23	50267	0.08
Time Pressure	3.16	2.97	42711	0.09
Economy	3.41	3.42	46738	0.973
Sensation	3.62	3.1	37094	1.17e-02
Classification	3.51	3.72	51340	0.042

Table 1 shows many significant differences ($p\text{-value} < 0.05$). Interestingly, Competition, Cooperation, Social Pressure, Scarcity and Classification were considered slightly more relevant by the males, whilst Renovation, Puzzle and Sensation elements were considered more relevant by females. Time pressure was disliked by males, but not as much by females.

4.2 Descriptive Analysis

Comparing surveyees, for elements preferred in different proportions, which are statistically significant between males and females, Cooperation was more relevant to males (57.3%); with 41.6% males selecting highly relevant, vs. 31.1% females (Table 2).

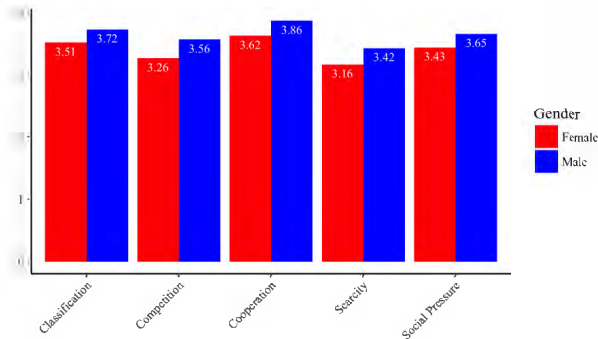


Figure 1: Favourite game elements for males (a,b-correspond to Tables 2,3, rsp.)

A more drastic difference appeared when more than 25% of the females did not consider Competition to be relevant, versus 54.5% males. This result suggests that males may perceive social interactions, such as Competition elements and Cooperation, as highly relevant overall in games, with a slight preference of Competition. For females, however, Competition is not that relevant, but, surprisingly, Cooper-

ation is only marginally relevant. Social Pressure is significantly less liked by females (difference of 7.7%).

Table 2: Cooperation, Competition and Social Pressure answers.

Sc	Cooperation				Competition				Social Pressure			
	F	%	M	%	F	%	M	%	F	%	M	%
1	12	7.3	37	6.5	20	12.2	42	7.4	14	8.5	32	5.6
2	20	12.2	53	9.3	25	15.2	80	14.1	26	15.8	63	11.1
3	38	23.8	100	17.6	44	26.8	137	24.1	45	27.4	146	25.7
4	43	26.2	142	25	43	26.2	135	23.7	34	20.7	160	28.1
5	51	31.1	237	41.6	32	19.5	175	30.8	45	27.4	168	29.5

Scarcity was instead favoured (significantly) by males (Table 3), where 37.8% of the females are indifferent. Classification, also a social element, was considered significantly more relevant by males. 50.6% of the females considered it relevant, against 60.6% males (Table 3). Based solely on our descriptive analysis, we observed that the male population considered limited or rare tasks, allowing, *e.g.*, rewards such as interaction or collecting titles, as relevant. Again, in practice, this information allows the teachers to create titles for completing specific tasks during their lectures *e.g.*, by giving a title of 'Speedster' to the student who completes a list of task correctly and quicker than the others.

Table 3: Scarcity and Classification answers

Sc	Scarcity				Classification			
	F	%	M	%	F	%	M	%
1	16	9.8	32	5.6	13	7.9	25	4.4
2	25	15.2	78	13.7	14	8.5	68	11.9
3	62	37.8	189	33.2	54	32.9	131	23
4	39	23.8	161	28.3	42	25.6	163	28.6
5	22	13.4	109	19.2	41	25.0	182	32.0

As for the elements most favoured by females (Figure 2), Renovation scored highest. 76.2% said it was relevant, with 50% considering it highly relevant. In contrast, only 30.0% of the males considered it highly relevant, with almost 30% indifferent (Table 4). In a learning context, this may tell the teacher that female students might be more pleased with features as "continue", "try again" or be given 'extra lives'.

Another element highly relevant to females was Puzzles: 80.8%, against 68.1% of males; with 21.9% males indifferent. Again, females, in this scope, considered that testing their skills was more relevant than males did. The Puzzle and Renovation elements, when combined in practice, allow problem solving, with the opportunity to correct mistakes.

Finally, the Sensation element was considered more relevant by females. More than half (54.8%) of the female sample considered it relevant, against 42.5% of the males. This could be explained by Sensation being related to the user experience [8], and, based on Table 4, we can infer that the most relevant elements for the female sample were related to the experience, rather than social ones. This means that they may perceive tasks that involve their senses, *e.g.*, with a visual or phonetic appeal, as more relevant, which could further be redone whenever they wish, to improve a certain skill through challenges. In practice, this means that using materials and resources that are more visual appealing may be more pleasant to female students than the male ones.

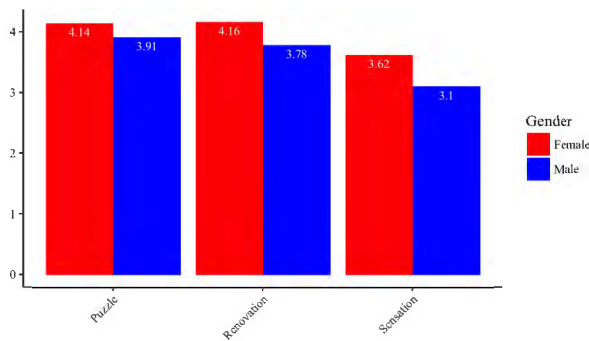


Figure 2: Favourite game elements for the female sample

Table 4: Renovation, Puzzle and Sensation answers

Sc	Renovation				Puzzle				Sensation			
	F	%	M	%	F	%	M	%	F	%	M	%
1	4	2.4	14	2.5	5	3	17	3	12	7.3	108	19
2	9	5.5	39	6.8	8	4.9	40	7	21	12.8	97	18
3	26	15.8	167	29.3	20	12.2	125	22	41	25	122	21.4
4	43	26.2	161	28.3	57	34.8	180	31.6	34	20.7	115	20.2
5	82	50	188	33	74	45.2	207	36.4	56	34.1	127	22.3

4.3 Association Rules analysis

To identify the strongest rules for each gender and to verify how the rules found matched or complemented the findings from our descriptive analysis (Section 4.2), we used the Apriori algorithm in Weka, with: (i) minimum support of 10% for the male sample size and 20% for female (to balance sample sizes); (ii) minimum confidence of 90%; and, after applying those attributes, (iii) we used the measures of interest *conviction*, *lift* and *leverage* to find the most interesting rules [23]. Using this setting we found a total of 11 rules for the female sample and 13 rules in the male one.

The majority (>90%) of the rules were based on the Objective element, which suggests its overall popularity. This translates into a general recommendation towards using 'Objective' elements in the educational gamification design, such as missions, milestones and quests, to guide the students. In this work we focused on analysing the most interesting rules in female and male samples. For females, the strongest rules were associated with the Objective (Table 5). The lift > 1 and leverage near 0 indicate that our items are independent and have a positive correlation, and the conviction between 1 and 5 indicates that these are interesting rules. The strongest rule relates Progress, Acknowledgement and Data elements (e.g., Representations of progression, badges and medals and results screen) with Objective (e.g., missions and quests). Rules regarding Progress and Level were also amongst the 10 strongest. Thus, we can suggest that teachers and instructors should use Acknowledgement (such as badges and trophies), with other elements associated with the personal enhancement of users (Progress and Level).

As for the male sample, Objective was also the main element but, in contrast to the females, we did not find any rules (with confidence > 90%) related to elements that were most relevant to the male population (Table 6). There was only 1 rule that specified a social element (Social Pressure)

amongst all the 14 rules. We can observe that Progress appears in almost all the rules, followed by Choice, appearing in seven rules. This means that, in our sample, designers and teachers should consider quests and missions that contain a form of progression and allow the students to make meaningful choices; those choices can be tied to a challenge (Rule 14), to transactions (Rule 24) and points (Rule 16).

Based on the data on Tables 5 and 6, we can observe that Objective associated to Progress is a concept that is (generally) well accepted by both genders. This means that teachers and designers should focus on, e.g., developing quests (which can be tied to their original learning objectives) that allow the learners to place themselves within the task. This is important, since in some educational context, students do not know why they are learning a specific content; and consequently, may become demotivated [24]. In practice, this means that teachers can create milestones or goals, allowing students to visualise their progress towards this goal. Thus, guidelines can be provided to teachers, to convert their objectives in their classes into milestones or quests. Additionally, other representations of Progress, showing the users where they are in the course could be implemented, such as those supported by Levels, Points and Data.

4.4 Further Discussion

We consider this work to be important, as, with the advent of 'big data', various theoretical assumptions and statements can now be backed up by (significant) evidence. In the case of game elements, there is firstly a vast (not always research-based) evidence that games are linked to motivation, and keep players 'in the flow' [25]. Some studies even link specific game elements to higher levels of commitment or motivation [9]. Based on this evidence, as well as theories of motivation, gamification has been proposed for education. Currently, however, the data supporting these assumptions is scarce. There is a lot of small-scale empirical evidence, at classroom-scale, of approaches that showed mixed successes [26, 27]. In a similar way, there is evidence that gamification can also have undesirable effects [28]. This clearly points to the fact that there are parameters which need taken into consideration, which may influence the outcomes of gamified approaches to education. In this study, we specifically focus on demographic parameters - namely, gender.

Gender in education has been brought to the fore recently, with the advent of initiatives such as the 'Athena SWAN'³ initiative towards gender equality in Higher Education in the UK, as well as similar initiatives world-wide. Importantly, equality doesn't mean 'one size fits all': on the contrary, gender equality means that the provision of education takes into account specific preferences that may be gender related. In a similar vein, certain types of games appeal to certain demographics and not others. For instance, card-related games are potentially more appealing to women, and first-player-shooter games to men (although, of course, preferences can vary) [29].

Further analysis of Table 1 shows that male and female preferences of some elements is relatively similar; e.g., Data, Economy, Objective are almost identical, and some are only

³<https://www.ecu.ac.uk/equality-charters/athena-swan/>

Table 5: Relevant association rules for female sample

Rule ID	If	Then	Conf	Lift	Lev	Conv
1	{progress, acknowledgement, data}	{objective}	0.97	1.63	0.08	7.04
2	{level, progress, acknowledgement}	{objective}	0.97	1.62	0.08	6.84
3	{progress, acknowledgement}	{objective}	0.96	1.6	0.1	6.04
4	{level, acknowledgement}	{objective}	0.95	1.59	0.09	5.5
5	{point, acknowledgement}	{objective}	0.95	1.58	0.08	4.96
6	{progress, puzzles}	{objective}	0.94	1.57	0.1	4.73
7	{puzzles, novelty}	{objective}	0.92	1.54	0.07	3.82
8	{novelty, acknowledgement}	{objective}	0.92	1.54	0.07	3.72
9	{acknowledgement, choice}	{objective}	0.92	1.54	0.07	3.72
10	{acknowledgement, data}	{objective}	0.91	1.52	0.08	3.54
11	{puzzles, acknowledgement}	{objective}	0.9	1.51	0.08	3.38

Table 6: Relevant rules to male sample

Rule ID	If	Then	Conf	Lift	Lev	Conv
12	{renovation, progress, choice}	{objective}	0.94	1.6	0.04	5.37
13	{progress, social pressure, data}	{objective}	0.93	1.59	0.04	5.04
14	{progress, puzzles, acknowledgement}	{objective}	0.93	1.59	0.05	5.16
15	{level, renovation, progress}	{objective}	0.93	1.59	0.04	4.96
16	{point, objective, puzzles}	{level}	0.92	1.93	0.05	5.74
17	{level, progress, puzzles, choice}	{objective}	0.92	1.57	0.04	4.27
18	{progress, acknowledgement, data}	{objective}	0.91	1.55	0.05	4.13
19	{point, progress, choice}	{objective}	0.91	1.55	0.04	4.01
20	{progress, novelty, data, choice}	{objective}	0.91	1.55	0.04	4.01
21	{progress, novelty, acknowledgement, choice}	{objective}	0.91	1.55	0.04	3.95
22	{renovation, progress, novelty}	{objective}	0.91	1.55	0.04	3.92
23	{level, progress, data, choice}	{objective}	0.91	1.55	0.04	3.92
24	{progress, novelty, economy}	{objective}	0.91	1.54	0.04	3.78
25	{progress, choice, economy}	{objective}	0.91	1.54	0.04	3.78

slightly different. Thus, some game elements may be perceived similarly by males and females - which makes the teacher's job much easier, in terms of design choices. This also puts more emphasis on the game elements where large differences exist, as well as on game elements where the differences in preference are slight, but statistically relevant.

Some of the results obtained were surprising: for instance, we expected females to appreciate cooperation more than males, but results (see section 4.2) showed otherwise. We did, on the other hand, obtain the expected results in terms of preference for competition. It is possible that online social interaction overall is perceived differently by males and females; for instance, females may perceive any type of social interaction online, where people are not known in advance, and anyone from anywhere can participate, as potentially threatening. These types of areas need further analysis.

For educational applications, it may seem that such potential 'fears' are less likely in controlled (classroom, or classroom-based) environments. However, for example, on Massive Online Open Courses (MOOCs), where people can participate from anywhere, such issues can again prevail. In fact, research on social interactions on MOOCs (e.g., comments, etc.) shows a predominance of males performing such activities. In contrast, females preferred puzzles (which can be solved also as solo-player) and the 'Renovation' element

(see Figure 2), which allow for an independent style of play where one focuses only on ones own progress, instead of being interrupted by others.

5. CONCLUSIONS

This work presents our approach based on DDGD, towards planning of gamification in the educational information systems domain by using data mining. The main contribution of our work is to present a poll of gamified strategies tied to male and female genders. Furthermore, we use real data to aid in the decision process of teachers and instructors is selecting gamification strategies. Through our data, we could identify that males would make more use of social interactions, with strong confidence rules pairing gamification elements Progression and Choice. For the females, we identified that user experience and rewards are more relevant, with association rules indicating a strong confidence for the need of Acknowledgement and Progression. We believe that this work can impact the way teachers perceive and apply gamification in their environments, consequently improving students' engagement and motivation through a game-like experience.

Acknowledgement

We would like to thank FAPESP (Projects 2016/02765-2; 2018/11180-3; 2018/07688-1) for the funding provided.

6. REFERENCES

- [1] S. Deterding, M. Sicart, L. Nacke, K. O'Hara, and D. Dixon, "From Game Design Elements to Gamefulness: Defining "Gamification"," *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems - CHI EA '11*, p. 2425, 2011.
- [2] K. Seaborn and D. I. Fels, "Gamification in Theory and Action: A Survey," *International Journal of Human-Computer Studies*, vol. 74, pp. 14–31, 2014.
- [3] J. Martí-Parreño, D. Seguí-Mas, and E. Seguí-Mas, "Teachers' Attitude towards and Actual Use of Gamification," *Procedia - Social and Behavioral Sciences*, vol. 228, pp. 682–688, jul 2016.
- [4] A. Sánchez-Mena and J. Martí-Parreño, "Gamification in higher education: teachers' drivers and barriers," *Proceedings of the International Conference of The Future of Education*, no. July, 2016.
- [5] A. M. Toda, R. M. do Carmo, A. P. da Silva, I. I. Bittencourt, and S. Isotani, "An approach for planning and deploying gamification concepts with social networks within educational contexts," *International Journal of Information Management*, oct 2018.
- [6] A. Mora, D. Riera, C. González, and J. Arnedo-Moreno, "Gamification: a systematic review of design frameworks," *Journal of Computing in Higher Education*, 2017.
- [7] A. van den Hurk, M. Meelissen, and A. van Langen, "Interventions in education to prevent STEM pipeline leakage," *International Journal of Science Education*, vol. 41, no. 2, pp. 150–164, jan 2018.
- [8] A. Dignan, *Game Frame*. Free Press, 2011.
- [9] N. Yee, "Motivations for Play in Online Games," *CyberPsychology & Behavior*, vol. 9, no. 6, pp. 772–775, dec 2007.
- [10] G. F. Tondello, R. R. Wehbe, L. Diamond, M. Busch, A. Marczewski, and L. E. Nacke, "The Gamification User Types Hexad Scale," in *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play - CHI PLAY '16*. New York, New York, USA: ACM Press, 2016, pp. 229–243.
- [11] L. Shi, A. I. Cristea, S. Hadzidedic, and N. Dervishalidovic, "Contextual Gamification of Social Interaction – Towards Increasing Motivation in Social E-learning," in *ICWL 2014*, vol. 8613, 2014, pp. 116–122.
- [12] L. Shi and A. I. Cristea, "Motivational gamification strategies rooted in self-determination theory for social adaptive e-learning," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9684. Springer, Cham, 2016, pp. 294–300.
- [13] E. L. Deci and R. M. Ryan, *Intrinsic Motivation and Self-Determination in Human Behavior*. Boston, MA: Springer US, 1985.
- [14] M. Denden, A. Tlili, F. Essalmi, and M. Jemni, "Does personality affect students' perceived preferences for game elements in gamified learning environments?" *Proceedings - IEEE 18th International Conference on Advanced Learning Technologies, ICAALT 2018*, no. August, pp. 111–115, 2018.
- [15] O. P. John and S. Srivastava, "The Big Five trait taxonomy: History, measurement, and theoretical perspectives," *Handbook of personality: Theory and research*, vol. 2, no. 1999, pp. 102–138, 1999.
- [16] J. Albuquerque, I. I. Bittencourt, J. A. Coelho, and A. P. Silva, "Does gender stereotype threat in gamified educational environments cause anxiety? An experimental study," *Computers and Education*, vol. 115, pp. 161–170, dec 2017.
- [17] L. Z. Pedro, A. M. Z. Lopes, B. G. Prates, J. Vassileva, and S. Isotani, "Does gamification work for boys and girls?" *Proceedings of the 30th Annual ACM Symposium on Applied Computing - SAC '15*, pp. 214–219, 2015.
- [18] I. Kotini and S. Tzelepi, "A Gamification-Based Framework for Developing Learning Activities of Computational Thinking," in *Gamification in Education and Business*. Cham: Springer International Publishing, 2015, pp. 219–252.
- [19] A. C. T. Klock, I. Gasparini, and M. S. Pimenta, "5W2H Framework," in *Proceedings of the 15th Brazilian Symposium on Human Factors in Computer Systems - IHC '16*. New York, New York, USA: ACM Press, 2016, pp. 1–10.
- [20] H. B. Mann and D. R. Whitney, "On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, jun 1947.
- [21] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur, "Dynamic itemset counting and implication rules for market basket data," *ACM SIGMOD Record*, vol. 26, no. 2, pp. 255–264, jun 1997.
- [22] M. Tavakol and R. Dennick, "Making Sense of Cronbach's Alpha," *International Journal of Medical Education*, vol. 2, pp. 53–55, 2011.
- [23] J. Manimaran and T. Velmurugan, "Analysing the quality of association rules by computing an interestingness measures," *Indian Journal of Science and Technology*, vol. 8, no. 15, pp. 1–12, 2015.
- [24] P. L. Hardré, "SUCCESS for teaching assistant professional development," *The Journal of Applied Instructional Design*, vol. 2, no. 1, pp. 50–55, 2012.
- [25] M. Csikszentmihalyi, *Flow: The Psychology of Optimal Experience*. Harper & Row, 1975.
- [26] S. d. S. Borges, V. H. S. Durelli, H. M. Reis, and S. Isotani, "A systematic mapping on gamification applied to education," in *Proceedings of the 29th Annual ACM Symposium on Applied Computing - SAC '14*, no. Icmc, 2014, pp. 216–222.
- [27] C. Dichev and D. Dicheva, "Gamifying education: what is known, what is believed and what remains uncertain: a critical review," p. 9, dec 2017.
- [28] A. M. Toda, P. H. D. Valle, and S. Isotani, "The Dark Side of Gamification: An Overview of Negative Effects of Gamification in Education," in *Communications in Computer and Information Science*, vol. 832. Springer, Cham, mar 2018, pp. 143–156.
- [29] Entertainment Software Association, "Essential Facts About the Computer and Video Games Industry," ESA, Tech. Rep., 2018.

Grading emails and generating feedback

Abhishek Unnam, Rohit Takhar, Varun Aggarwal
Aspiring Minds
{abhishek.unnam, rohit.takhar, varun}@aspiringminds.com

ABSTRACT

Email has become the most preferred form of business communication. Writing *good* email has become an essential skill required in the industry. *Good* email writing not only facilitates clear communication, but also makes a positive impression on the recipient, whether it be one's colleague or a customer. The aim of this paper is to demystify the components of a *good* email and to define a set of parameters by which to grade the quality of an email and provide detailed feedback. This can help candidates improve their email writing skills and also guide tutors. The email grading parameters encompass traditional attributes of written English (i.e. coherent and relevant content and correct grammar) but also include a unique set of characteristics that we may objectively identify as email etiquette. These characteristics comprise the metrics we use to evaluate the quality of the various constituent parts of an email. We grade the email using artificial intelligence, acting on semi-structured text. We use a mix of machine learning and rule-based systems to effectively grade an email on the specified parameters. Our system automatically grades email with accuracy comparable to human graders.

Keywords

Automatic grading; Email Writing; Semi-Structured Text analysis; Education; Supervised Learning

1. INTRODUCTION

In today's knowledge economy, good communication skills (spoken as well as written) are vital for success in the workplace. According to the O*NET taxonomy of jobs and skills [5], 53% of all jobs require a moderate to high level of writing and speaking skills. Lately, there has been a lot of work [18], [10], [17] on automated evaluation of speaking skills. Some of these automated systems help candidates to improve their language-speaking skills [3], [7]. The evaluation of writing skills is generally thought to be confined to academic essay grading [1], [14]. These studies were prompted

by the demand for more efficient evaluation of high-volume educational/academic tests such as TOEFL and SAT. There are job tasks that mimic essay writing such as writing user manuals, product documents or filing Request-for-Proposals (RFPs). However, the writing that is most ubiquitous in companies is email. Interactions with clients and all manner of internal communications with managers, peers, and support services (i.e. human resources, tech support, etc) are carried out via email. Therefore, composing *good* email has become a necessary skill.

We wish to automatically grade email writing skills and provide feedback. This will create a mechanism for students and jobseekers to get feedback on their email writing skills and also provide companies with the ability to automatically test email writing skills of job candidates on a large scale and use the grades in the hiring process. We wish to automatically grade email writing skills and provide feedback. This will create a mechanism for students and jobseekers to get feedback on their email writing skills and outline a path for improvement. It will also provide companies with the ability to automatically test email writing skills of job candidates on a large scale and use the grades in the hiring process. Despite the importance of email writing skills, we have not found any previous work in this area.

We first demystify the components of a *good* email. The grading parameters consist of the traditional attributes of written English (i.e. coherent and relevant content and correct grammar) but also include a unique set of characteristics that we objectively identify as email etiquette. We have identified a set of 36 metrics on which to evaluate the quality of the various constituent parts of an email. Many of these metrics are derived from the general norms of communication, while others are specific to the written form of email (discussed in detail in Section 2). Our aim is to automatically grade email on each of these metrics.

We grade email using artificial intelligence, acting on semi-structured text. Some parts of an email are structured, such as the subject, salutation and sign-off (Figure 1). The body of the email, a set of sentences, is unstructured. There are multiple approaches to grading structured and unstructured text. Essay grading [4] and the grading of short answers [13] are examples of unstructured text grading. Generally, researchers generate a set of features such as bag of words, word embeddings, parts-of-speech (POS) tags, etc. and use supervised or semi-supervised learning to predict grades. The grading of computer programs which use tightly defined grammar is an example of structured input grading. In [15] authors derive sophisticated features by exploiting the

Abhishek Unnam, Rohit Takhar and Varun Aggarwal "Grading emails and generating feedback" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 444 - 449

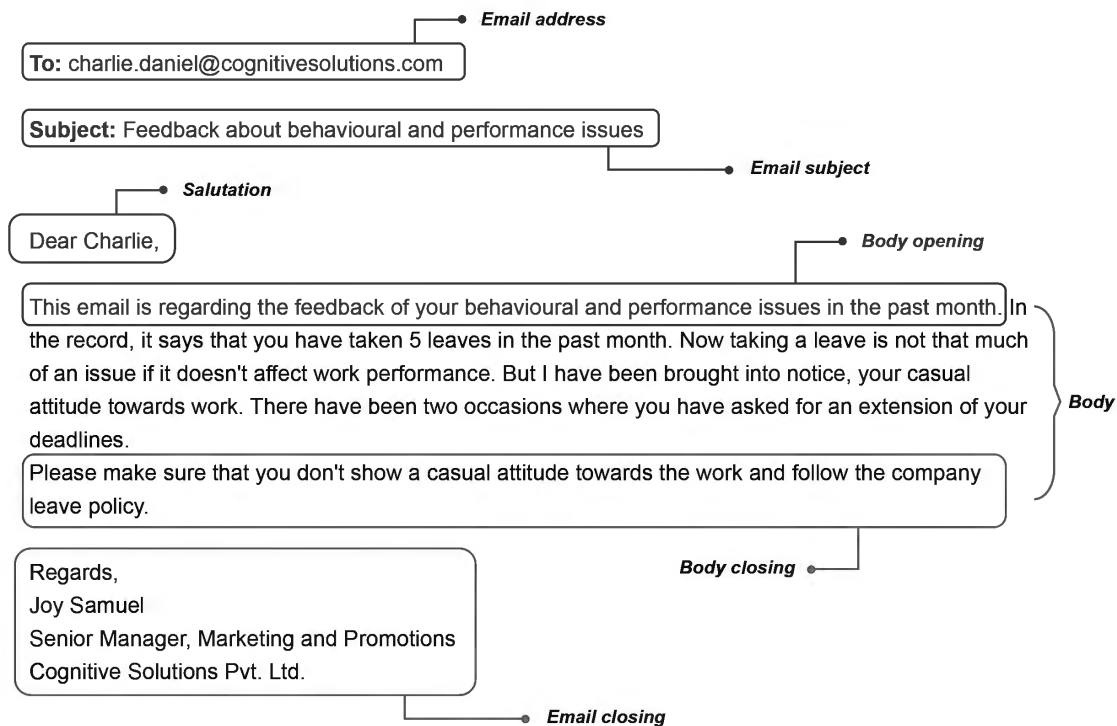


Figure 1: Sample email being written by a student.

structure of the program by drawing control row and data dependency graphs. Whereas, resume parsing is an example of processing semi-structured text [2].

We propose a mix of machine learning and rule-based models to grade the various quality metrics of an email. We find that such an approach works well for semi-structured text and for a variety of evaluation parameters. We show that generally the rule-based models work well for processing the structured parts of email and for grading email on etiquette. On the other hand, content relevance and grammar are better graded via machine learning. Interestingly, for some attributes, a mix of both models work best. Our approach stands contrast to recent trends that apply machine learning to all tasks indiscriminately.

We derive a number of natural language features for our machine learning models. We use supervised learning to build the models. The rule-based models use word lists and regular expressions. Our algorithm provides accuracy that rivals expert consensus. Our final system generates separate grades for content relevance, grammar and email etiquette. It also provides detailed feedback on the types of errors that occur in each part of the email. In particular, the paper makes the following contributions:

- We propose the first viable system for automatically grading email writing skills and delivering constructive feedback.
- We demystify the components of *good* email and provide objective evaluation criteria.
- We propose a mix of machine learning and rule-based models for evaluating emails. The accuracy of our system rivals expert consensus.

- We provide first-of-its-kind data-based insight into the errors committed by students and jobseekers.

The paper is organized as follows. In Section 2, we describe the metrics we define for evaluating email etiquette. In Section 3, we discuss the methods used to evaluate the metrics. In Section 4 we explain the experiments, datasets and models that have been developed for grading. In Section 5 we present our results by comparing the performance of our system with that of human experts. Concluding remarks are provided in Section 6.

Response	Status
I care for this in an email that I receive	agree
I may care for this in an email that I receive	not sure
I do not care for this in an email that I receive	disagree

Table 1: Each rule was put into one of these categories

2. EMAIL ETIQUETTE

Quality metrics for evaluating email writing skills go beyond the parameters traditionally defined in the grading of written language, i.e. paragraphs and essays.

Email as social communication must follow certain norms. Some of these norms are derived from the general norms of verbal communication, while others are specific to the written form of email. These norms manifest themselves as rule sets for the structured fields of email and the unstructured body. A simple rule is that values for all the structured fields should be present. For example, an email that lacks a subject line is

Quality Metrics	Explanation & Examples	Counts
Missing	Missing subject line, salutation, signoff etc. Subject: Email Body: This is to inform you that....	3
Redundancy	Starting the subject line with terms such as ‘regarding’, ‘response to’. Subject: regarding behavioral and performance issues.	6
Word usage	Incorrect usage of words in various sections of an email - using names/greetings in subject line, usage of informal, abbreviated words etc. Subject: Employee feedback for Charlie Daniels Email Body: Hi Daniel, Can u pls respond quickly...	10
Style	Errors specific to conventions like greeting and sign-off style. Email Body: Hella Daniel/Heyy Charlie Daniels/Hi Mr Charlie Daniel Yours Sincerely/Truly/faithfully Mr Charlie Daniel	7
Emotional Punctuation	Errors like using too many commas inside a sentence, using exclamation/semi-colons marks inside subject/salutation/closing, using all uppercase words in subject line etc. Subject: POOR performance !!! need improvement Email Body: Heyy!! Charlie, HI CHARLIE please reply...Thanks, Alisha	5
Punctuation	Capitalisation errors like starting subject line with lowercase, proper nouns starting in lowercase etc. Not giving space after fullstop. Subject: feedback on performance Email Body: hi charlie daniel, This is to inform you about the poor performance in last financial year.I have seen many instances of work lapse.	5

Table 2: Quality metrics with explanations & examples.

not good. A rule derived from verbal communication norms is that the opening greeting should not address the recipient by both first and last name, and that a title should be added when addressing a person by last name only. An example of a rule typical to the written form is that the subject line should not be longer than a given length. Another example is that no words (except acronyms) should appear in all upper case, and that emoticons should be avoided. There are also other metrics that derive from the changing norms of communication. In today’s business world, one doesn’t address others as ‘Respected’ and rather use a ‘Hi/Hello/Dear’. There are similar rules that govern the body of an email.

There are additional regarding the purpose of the email, or in a grading scenario, the prompt of the email writing task. For example, when responding to an irate customer, one should not employ the oft-used email phrase “Hope you are doing well”. Similarly, the closing line of an email may depend on the prompt. Emails that seek a response may conclude with, “I look forward to your response”, while a simple conversation may end with simply “Feel free to reach out to me”. There is no standard list of these rules. We looked at the various blogs and documents with email writing rules to put together a super-set of 57 different possible rules. We shared these rules with three professionals from India and three from US, each with 10+ years experience in the corporate world. Two from each country were from either the IT, banking or contact center industry. Three were involved in external communications, while another three were into internal com-

munications. We asked them to rate each rule as in Table 1. From the original 57 rules, we selected the 36 on which four

Section	Number of errors
Subject	9
Salutation	6
Email Body	13
Closing	8

Table 3: Section-wise error counts.

or more of the professionals agreed. These 36 rules became our quality metrics, which we categorized into 6 broad categories with definitions (See Table 2). In Table 3, we show the number of rules that apply to each part of an email. We evaluate an email against each of these rules and delineate the errors in a candidate feedback section. We also provide a total score for email etiquette. This score is based on the number and severity of errors made. For example, a *missing* error is more severe than *redundancy* errors. We also include the standard parameters and rubrics [8] of English evaluation: content relevance and grammar.

3. METHODS USED

We used three different methods to evaluate the various facets of quality metrics. We use a mix of rule-based and machine learning methods.

3.1 Word list-based

Many email etiquette errors may be detected simply by presence or absence of certain words or phrases. For instance, an email should not have abusive words or slangs. On the other hand, a pleasing email shall have words like ‘thank you’, ‘please’, ‘request’ etc. We created word-lists to evaluate such metrics in an email. Such an approach has also been followed earlier in emotion detection [16]. The words in email are matched against the word lists after stemming. Based on the occurrence and counts (normalized, in some cases), the feedback and scores, are respectively generated. For instance, if someone writes “Whoa, It was great working with u.” we provide a feedback to avoid the usage of informal words like ‘whoa’ and ‘u’. We graded 19 error metrics in this manner using 26 word lists¹. The efficacy of this approach was tested by evaluating the generated feedback and score against human expert feedback. This is described in Section 4.

3.2 Pattern-based

Certain metrics were evaluated not by the occurrence of a single word or phrase, but based on an expected pattern. For instance, the recipient may be addressed by using more than one combination of the greeting (Dear/Hi/Hey), title (Mr/Ms/Dr), first name and last name. Some combinations are right (Dear First Name), while some are wrong (Dear Last Name). The correct and incorrect patterns were coded into regular expressions, to provide exact detection of an error and providing feedback (e.g., ‘Do not use only last name without title’). One may note that this detection assumes we know the full name of the recipient. For every email writing task (prompt), certain structured information such as recipient name, company name, sender name and certain keywords is generated at the beginning (manually). This is used for various error detection including, for example, capitalization, and spelling errors. In total, 16 error metrics were evaluated using pattern matching.

3.3 Regression-based

We use supervised learning to grade an email on content relevance and grammar. We train regression models using a number of different features to predict expert grades. We now describe the features used for each of the two parameters.

3.3.1 Content Relevance

The candidate’s task was to write an email according to the situation provided in the prompt. We wish to evaluate whether the content properly addresses the situation, is comprehensive, coherent and without unnecessary information. This metric is linked to the semantics of the email only and doesn’t evaluate other parameters such as the emotion of the email. We describe the natural language features used for the task.

- **Word embeddings:** Here, we used the Word2vec model [11], particularly Google’s pre-trained model developed with a vocabulary over 3 million words and phrases and trained on roughly 100 billion words from Google’s news dataset. For each word in the email we first calculate a 300-length lower dimensional vector and then sum it across all the words in the email.

¹For some error metrics more than one word lists were used.

- **Bag of Words (BOW):** We used the bag of words feature-counts of unigrams, bigrams and trigrams. All the words were stemmed and stops words were removed.
- **Prompt Overlap:** We calculated prompt overlap in two ways: *exact match* and *extended match*. In *exact match*, we count the number of common words between the prompt and the email. In *extended match*, we add the synonyms of all words in the prompt using WordNet [12]. We then count the number of common words between the extended prompt word list and the words in the email.

3.3.2 Grammar

Below are the features we used to evaluate the grammatical correctness of Here our aim is to evaluate the grammatical correctness of the email. We use the following features:

- **Bag of POS tags:** Here, words are assigned to their respective part of speech (POS) tags using the Penn Treebank NLTK tagger [9]. We then considered bigrams and trigrams of POS tags. This feature removes the semantic information from the words, while preserving the sentence structure and grammatical features.
- **Error Counts:** We also use counts of the grammatical errors in the email as identified by open-source grammar correction tools.
- **Proportion of good tags:** Here, we wish to find the similarity of the language in the email with that of a grammatically correct corpus. We used Brown corpus [6] for our purpose. We generated bag of POS bigrams and trigrams from this corpus. We consider the top 70% most frequently occurring POS bigrams and trigrams as a set of *good n-grams*. We then find what proportion of n-grams in the email are *good n-grams*.

4. EXPERIMENTS

We had designed our experiments to answer the following questions:

- How accurate is our approach in predicting content and grammar scores as compared to human experts?
- How accurately do we detect email etiquette errors using the word list and the pattern matching methods?
- What proportion of errors marked by humans experts do we correctly detect and how many false errors do we generate?

We conducted our experiments on a set of 1200 emails which were manually rated by human experts. For training models, we made use of both linear (linear, ridge regression) and non-linear (random forest(R.Forest), SVM) techniques. We discuss more about the dataset in the next section.

4.1 Dataset

Our dataset consists of 1200 emails in response to three different prompts. We used an equal set of 400 emails per prompt, after removing any blank email or those with very little content. These samples were collected from both undergraduate and graduate students. The candidates were given a situation and asked to write an email to address the situation. The three situations included a customer service situation where one needs to address a customer’s complaints, a sales situation where one probes the requirement of a prospect and

		Prompt 1		Prompt 2		Prompt 3	
Model	#Features	Train (r)	Validation (r)	Train (r)	Validation (r)	Train (r)	Validation (r)
Linear	50	0.86	0.79	0.87	0.81	0.85	0.77
Ridge	50	0.86	0.80	0.87	0.81	0.84	0.77
R.Forest	150	0.95	0.80	0.95	0.82	0.92	0.79
SVM	50	0.84	0.79	0.86	0.80	0.83	0.76

Table 4: Performance of prompt-specific content models.

		Validation				
Model	#Features	Train (r)	Overall (r)	Prompt 1 (r)	Prompt 2 (r)	Prompt 3 (r)
Linear	75	0.77	0.62	0.59	0.65	0.65
Ridge	75	0.70	0.66	0.62	0.67	0.75
R.Forest	150	0.85	0.73	0.71	0.73	0.74
SVM	150	0.62	0.60	0.52	0.59	0.74

Table 5: Performance of grammar models trained on complete set of emails. We present the overall and prompt wise validation correlations(r).

promotes a service and a people management situation where one needs to give feedback to an employee on performance issues. The responses were collected using *AMCAT*², our proprietary computer based testing platform.

All email responses were graded by human experts. Content Relevance and Grammar were graded based on a 7 point and 5 point rubric respectively. Detailed guidelines were provided for identifying each email etiquette error. The experts had to mark the exact location where the error occurred and the category of the error. Each email was graded by 3 different experts. These included an English language trainer, a sales manager and a customer service manager. Each of these had an experience of more than seven years in the industry. The experts first went through a three-day training where they learned how to interpret the rubric and were subjected to practice grading exercises.

We achieved an average inter-rater correlation 0.83 for content scores and 0.74 for the grammar scores. For email etiquette, only those errors were considered where atleast two experts had a consensus. A consensus was reached for 83% of the total cases.

4.2 Models

For content relevance we trained different models for each prompt, while for grammar, a generic model was trained across all prompts. For each model the corresponding dataset was divided into train and validation sets. We used a stratified 70-30 split for train-validation sets. We used linear regression, linear regression with L2 regularization (ridge), SVM and R.Forest to train models. Select K-best algorithm was used for feature selection. The models with the lowest cross-validation (4-fold) error were selected.

For the rule-based system, we used 50% of the email sets to experiment with the rules, patterns and word-lists. Several iterations were performed to fine tune the algorithm. The rules were then tested on the remaining 50% data. We report the results on this validation set.

²<https://www.aspiringminds.com/contentTech>

5. RESULTS

We evaluate our machine learning models using the Pearson correlation coefficient between predicted and the expert grades. We report and discuss results for the validation set. For content evaluation, all modeling techniques provide similar results. The correlation for all prompts is around 0.79 (refer Table 4). For grammar scores, *R.Forest* gives the best results, though marginally better than other techniques. Here, we created one model across prompts and the overall correlation across prompts is 0.73. Also, the correlation value of any individual prompt is more than 0.71 (refer Table 5). For email etiquette errors, we report two metrics of errors:

Error Category	Counts(%)	TP(%)	FP(%)
Redundancy	31.27	70.00	17.86
Missing	69.77	93.55	11.29
Punctuation	90.78	94.12	19.17
Emotional Punctuation	18.00	84.48	6.90
Style	95.74	95.11	8.27
Word usage	17.05	100.00	20.00
Average	53.76	89.54	13.91

Table 6: Category wise performance of rule based system for email etiquette.

- **TP (True Positives):** It is the proportion of expert errors that were correctly identified. An error is deemed correctly identified only when the position and error type is correct.
- **FP (False Positives):** It is the number of non-existent errors identified, normalized by the total count of expert errors. This provides an idea of the proportion of extra errors detected.

We report all the values for the unseen validation set, 50% of the total data. We present the category-wise and section-wise results in Table 6 and Table 7 respectively. **Counts(%)** (in Table 6 and 7) states the percentage of emails that had the given error. This provides some interesting insights. For

Section	Counts(%)	TP(%)	FP(%)
Subject	70.41	70.59	8.82
Salutation	79.84	92.78	17.53
Closing	94.88	95.28	11.02
Email Body	77.43	89.83	10.10

Table 7: Section wise performance of rule based system for email etiquette.

instance, we find that most students make an error in the email closing, followed by salutation and the least number of errors in writing the subject line. On the other hand, the most number of emails are impacted by errors in style and classical punctuations. When we analysed in detail, we find that the *style* of the *closing* has the most number of errors. Candidates either completely miss the sign-off or use overly formal phrases like ‘Yours Truly/Sincerely/faithfully’. We find that the average TP is 89.54% and FP is 13.91%. One may recall that the expert consensus was achieved in 83% of the cases. Our system detects most of the expert-identified errors and has a low rate of detecting false errors. Further, we find that the lowest TP rate for any category is 70% (*redundancy*) and highest FP rate is 20% (*word usage*). There are only two categories with either a TP rate less than 80% or a FP rate of more than 20%. On the other hand, the lowest TP is for the *subject* (70.6%) and highest FP is for *salutation* (17.5%) yet under the 20% mark. We ultimately aim to get all error rates in the 80-20 range of TP-FP.

6. CONCLUSION

We propose a system to grade email on content relevance, grammar and email etiquette using a mix of rule based and machine learning methods. We present a set of 36 quality metrics to evaluate email etiquette, their broad categorization and explanation. Our automated system provides human competitive performance on all evaluation parameters. The system provides scores on the three parameters and also detailed feedback on email etiquette. This feedback comprises the exact location of error, details of the error, and possible corrections.

In future, we plan to work on grading the finer aspects of email writing skills, such as flow of the email, its sentiment and how well the different parts of the email address the situation in the prompt. Also, this is a new interesting approach to process semi-structured text. We plan to use and benchmark the approach for other applications.

7. REFERENCES

- [1] Yigal Attali and Jill Burstein. Automated essay scoring with e-rater v.2.0. *Journal of Technology, Learning, and Assessment*, 2006.
- [2] Z. Chuang, W. Ming, L. C. Guang, X. Bo, and L. Zhi-qing. Resume parser: Semi-structured chinese document analysis. In *2009 WRI World Congress on Computer Science and Information Engineering*, volume 5, pages 12–16, March 2009.
- [3] Maxine Eskenazi, , Maxine Eskenazi, and Scott Hansma. The fluency pronunciation trainer. In *Proc. Speech Technology in Language Learning 1998, Marholmen*, pages 77–80, 1998.
- [4] Peter W. Foltz, Darrell Laham, and Thomas K Landauer. Automated essay scoring: Applications to educational technology. In Betty Collis and Ron Oliver, editors, *Proceedings of EdMedia + Innovate Learning 1999*, pages 939–944, Seattle, WA USA, 1999. Association for the Advancement of Computing in Education (AACE).
- [5] National Center for O*NET Development. O*NET Resource Center. <https://www.onetcenter.org/overview.html>. Accessed: 2019-02-28.
- [6] W. N. Francis and H. Kucera. Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979.
- [7] Horacio Franco, Harry Bratt, Romain Rossier, Venkata Gadde, Elizabeth Shriberg, Victor Abrash, and Kristin Precoda. Eduspeak(r): A speech recognition and pronunciation scoring toolkit for computer-aided language learning applications. *Language Testing*, 27, 08 2010.
- [8] Yong-Won Lee, Claudia Gentile, and Robert Kantor. Analytic Scoring of TOEFL CBT Essays: Scores from Humans and E-rater, 06 2008.
- [9] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP ’02, pages 63–70, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.
- [10] Angeliki Metallinou and Jian Cheng. Using deep neural networks to improve proficiency assessment for children english language learners. In *INTERSPEECH*, 2014.
- [11] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.
- [12] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.
- [13] Michael Mohler, Razvan C. Bunescu, and Rada Mihalcea. Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *ACL*, 2011.
- [14] Lawrence M. Rudner and Tiankai Liang. Automated essay scoring using bayes’ theorem. *Journal of Technology, Learning and Assessment*, Vol. 1, No. 2., 2002.
- [15] Shashank Srikant and Varun Aggarwal. A system to grade computer programming skills using machine learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1887–1896. ACM, 2014.
- [16] Yla R. Tausczik and James W. Pennebaker. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of Language and Social Psychology*, 29(1):24–54, 2010.
- [17] Zhen Wang, Klaus Zechner, and Yu Sun. Monitoring the performance of human and automated scores for spoken responses. *Language Testing*, 35(1):101–120, 2018.
- [18] Klaus Zechner, Derrick Higgins, Xiaoming Xi, and David M. Williamson. Automatic scoring of non-native spontaneous speech in tests of spoken english. *Speech Communication*, 51:883–895, 10 2009.

Improving Peer Assessment Accuracy by Incorporating Relative Peer Grades

Tianqi Wang
University at Buffalo
twang47@buffalo.edu

Qi Li
University of Illinois
qli22@buffalo.edu

Jing Gao
University at Buffalo
jing@buffalo.edu

Xia Jing
Tsinghua University
jingxyy@qq.com

Jie Tang
Tsinghua University
jery.tang@gmail.com

ABSTRACT

Massive Open Online Courses (MOOCs) have become more and more popular recently. These courses have attracted a large number of students world-wide. In a popular course, there may be thousands of students. Such a large number of students in one course makes it infeasible for the instructors to grade all the submissions. Peer assessment is thus an effective paradigm that can help grade the submissions at a large scale. However, due to the variance in the ability and standard of the student graders, peer grades may be noisy and biased. Aggregating peer grades to have an accurate and fair final grade for a submission is a challenging problem because the reliability and bias degrees of graders are usually unknown in practice. To address this issue, some probabilistic models considering the graders' reliability and bias are proposed. However, due to the sparsity of peer grade observations, it is difficult for these models to estimate the accurate reliability and bias of the graders as well as the true grades of the submissions. Compared with absolute peer grades, the relative peer grades, derived from the difference between the peer grades of two submissions graded by the same grader, are less sparse and more robust to the grader's bias. Thus relative peer grades are informative and helpful in cardinal peer grading estimation whose goal is to estimate the absolute numeric grades of submissions. In this paper, we propose two new probabilistic models to help improve the accuracy of cardinal peer grading estimation using the observed relative grades among submissions. In this way, the relation between the true grades among submissions is taken into consideration when deriving the final grades. Experimental results on real MOOC peer grading datasets show that the proposed models outperform baselines and the relation of true grades among submissions indeed contributes to the improvement in the grade estimation.

Keywords

Peer grading, relative peer grades, MOOCs

Tianqi Wang, Qi Li, Jing Gao, Xia Jing and Jie Tang "Improving Peer Assessment Accuracy by Incorporating Relative Peer Grades" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 450 - 455

1. INTRODUCTION

Massive Open Online Courses (MOOCs) have provided millions of learners with open access to high quality courses via web. For a popular course, there may be thousands of students. Recently, several MOOC platforms offer verified certificates or even degree programs, and peer grading plays an important role in the student performance evaluation. The benefit of peer grading is two-folded. On one hand, it is helpful for the instructors to evaluate students performance, which is otherwise infeasible due to the large number of enrollment. On the other hand, it is also beneficial to the students: they can see peers' work from different aspects and increase their involvement in the course [5]. Especially, peer grading can be used when automatic grading cannot be applied, for example, on essays and projects. A typical process of peer assessment includes two steps: first, students are assigned to grade a subset of submissions and then the platform aggregates these peer grades to compute the final grades of these submissions.

Although peer grading is helpful, it is a challenging problem to aggregate these peer grades and determine the final grade of a submission. In this paper, we consider the case of cardinal peer grading (i.e., each submission receives a numerical grade as the final grade). Most platforms use the median of received peer grades as the final grade of a submission. However, the median grade may be inaccurate due to the different reliability and bias degrees of graders. Usually, the difference between the grade given by a grader and the true grade of the submission can be decomposed into bias and reliability degree. Suppose a grader grades multiple submissions, and then the bias represents the difference between the mean grades of this grader and the true grades on these submissions. The reliability degree of the grader is measured by the variance of the difference between the the grades that the grader gives and the true grades of these submissions. If a grader randomly assigns grades to the submissions, he/she is not a reliable grader. If the variance is small, then a grader grades the submission in a consistent way and is thus a reliable grader. It is important to consider the modeling of grader bias and reliability to derive more accurate estimates of the final grades. Therefore, there are some existing efforts towards this direction [7].

However, the mechanism of peer grading that each student only grades a small subset of submissions leads to a data sparsity issue. The sparsity of the observed grades makes these models difficult to correctly estimate reliability, bias

of the grader and the true grades of the submissions. In addition the observed grades are sensitive to the grader's bias. Compared with absolute observed grades, the relative peer grades between two submissions are less sparse and more robust to the grader's bias, since the relative peer grades are derived from the difference of the grades assigned by the same grader to two different submissions. Thus the relative peer grades are informative in estimating the true grades of submissions. However, all existing cardinal peer grading estimation models [7, 6, 2] only consider the absolute peer grades of each submission. None of these models considers the relative grades between two submissions.

Recognizing the importance of relative peer grades, we develop new probabilistic graphical models by leveraging relative peer grades between submissions to model the dependency between the true grades. The proposed probabilistic models estimate the true grades of submissions from the peer grades as well as relative peer grades by modeling the bias and reliability of graders. Gaussian distributions are applied to model the true grades, the bias of grader, the absolute peer grades, and relative peer grades in the proposed models. Two different distributions are proposed to estimate the reliability of the graders. In the first model, the reliability of the grader follows a Gamma distribution with the shape parameter determined by the grader's own true grade, while in the second model, it follows a Gaussian distribution with the mean equal to the grader's true grade. To evaluate the proposed models, experiments are conducted on peer grading datasets collected from a popular MOOC platform in China. Experimental results show that the proposed models improve the accuracy of the cardinal peer grading estimation by considering the dependency of true scores between two submissions. The main contributions of this paper are summarized as follows:

- We find that relative peer grades among submissions can help improve cardinal peer grading estimation accuracy.
- We propose new probabilistic graphical models by incorporating observed relative grades to model the dependency between the true grades of these two submissions.
- We evaluate the proposed models on real peer grading datasets and experimental results show that the proposed models can improve the accuracy of cardinal peer grading estimation.

2. RELATED WORK

Existing work on peer assessment aggregation can be divided into two categories based on the data types: the cardinal and ordinal peer grade estimation. The goal of ordinal peer grade estimation is to rank the students according to their submissions. Models based on pair comparison [10, 8], Bayesian generative approach [12] and matrix factorization are developed for the ordinal peer grades estimation [1].

For cardinal peer grading estimation, students are asked to grade their peers' submissions by assigning a specific numerical grade and the aim of cardinal grades estimation is to find the absolute true scores of the submissions. Below we summarize the existing work related to cardinal peer grading estimation respectively.

One major approach of cardinal peer grading estimation is to

update grades and grader weights iteratively [4, 12, 3]. Another major category of methods are based on probabilistic graphical models [7, 6, 2]. The proposed models in this paper fall into this category. The main idea is to model the true grade of a submission, the reliability and bias of each grader as hidden random variables following certain distributions, and infer the model parameters by fitting the models on observed peer grades. In particular, the following methods [7, 6] (referred to as PG_1 to PG_5) are the most relevant to our proposed model. In [7], three probabilistic graphical models named PG_1 , PG_2 and PG_3 are proposed. PG_1 is the basic model, which assumes that true grades, observed peer grades, and biases follow Gaussian distributions and the reliability of the grader follows a Gamma distribution. Upon PG_1 , PG_2 links the bias of a grader among assignments, and PG_3 couples the grader's grade of his/her submission and the grader's reliability. In PG_3 , the grader's reliability is modeled as a linear function of the grader's grade. To relax this assumption of linear relationship, two extensions of PG_3 referred as PG_4 and PG_5 are later proposed in [6]. Both PG_4 and PG_5 assume the reliability of a grader is related to the grader's own grade, and use either Gamma distribution or Gaussian distribution to model this reliability. Recently, social connections are also considered in the modeling of the dependencies of bias among students [2].

However, all existing cardinal peer grading estimation methods only consider absolute grades. In these methods, the true grades of different submissions are treated independently. None of these models takes the relative grades into consideration. In fact, leveraging the relative grades between submissions to model the dependency between true grades of these two submissions can help reduce the noise introduced by the bias of graders and alleviate the data sparsity issue, and thus can help to improve the accuracy of cardinal peer grading estimation. To the best of our knowledge, this is the first work that integrates relative grades into cardinal peer grading aggregation to achieve improved estimation.

3. PROBLEM DEFINITION

In this section, we first introduce some concepts and notations used in the rest of this paper. Then we formally define the problem.

The set of all the students is denoted as S and the set of all the graders is denoted as G . Under the peer grading setting, $G \subseteq S$, since the graders are students as well. The observed absolute grade (peer grade) of a submission submitted by student i graded by grader g is denoted as z_i^g , and the observed relative grades (relative peer grades) between submissions submitted by students i and j graded by grader g is denoted as d_{ij}^g . The relative peer grades are derived using absolute peer grades, which are the difference of the absolute peer grades. For example, if a grader g assigned a score of 4 to the submission submitted by student i and a score of 6 to the submission submitted by student j , then z_i^g is 4 and z_j^g is 6. We can derive that the relative grade $d_{ij}^g = z_j^g - z_i^g = 6 - 4 = 2$. The subset of students whose submissions are graded by an arbitrary grader $g \in G$ is described as S_g and the set of graders who assign grades to the submission submitted by student i is defined as G_i .

With these definitions introduced, we define the cardinal peer grading estimation problem as follows: Given a set of

students S , a set of graders G , a set of peer grades $\{z_i^g\}_{i \in S, g \in G}$ and relative peer grades $\{d_{ij}^g\}_{i, j \in S, i \neq j, g \in G}$, we want to estimate the true absolute grade for submission submitted by student i , $\forall i \in S$, and to learn the reliability and bias for each grader g , $\forall g \in G$.

4. METHODOLOGY

In this section, we describe our probabilistic graphical models named PG_6 and PG_7 for cardinal peer grading estimation. Both models specify a two-stage generation for the peer grades and relative peer grades. The first stage specifies the generation of graders' bias, reliability and true scores of submissions and the second stage generates the peer grades and relative peer grades given the grader's bias, reliability as well as the true scores of submissions.

True score generation: In the proposed models, the true score of the submission submitted by student i is modeled as a random variable following a Gaussian distribution.

Grader bias generation: The bias of grader g is denoted as b_g , which measures the constant grade inflation or deflation of a grader. We model the grader's bias as a random variable following a Gaussian distribution. Though different graders may have different bias, we can assume the average of all graders' bias is 0.

Grader reliability generation: The reliability of a grader reflects how consistent a grader assigns grades. A reliable grader keeps a stable bias when assigning grades to different submissions. Following the assumptions in [11], we assume that the reliability of a grader is related to his/her own grade, which reflects the grader's knowledge about the assignment. We assume that the grader with a higher grade of the assignment may be a more reliable grader for submissions of the same assignment. The reliability of a grader g is denoted as τ_g and modeled as a random variable following a Gamma distribution in the PG_6 model and a Gaussian distribution in the PG_7 model, respectively. In the PG_6 model the grader's true grade is used as the shape parameter of the Gamma distribution, while in the PG_7 model it is used as the mean value of the Gaussian distribution.

Peer grade generation: After generating the bias and reliability of graders as well as the true scores, the peer grades can be generated with these variables. The peer grade is modeled as a variable following a Gaussian distribution whose mean is the sum of the true grade of the submission and the bias of the grader, and its variance is inversely proportional to the reliability of the grader. In the PG_7 model, we introduce a hyper-parameter λ to tune the scale of the variance.

Relative peer grade generation: To incorporate more observations to estimate the reliability and bias of the grader and the true grade of the submission, the relative peer grade is generated. The generation of relative peer grade provides us with another view of true score of a submission in addition to the traditional way that models the true grade as the sum of observed peer grade and the bias of the grader. With the relative peer grade, the true grade s_i of submission i can be estimated by the sum of the true grade s_j of submission j and the relative peer grade between these two submissions. In such a way, the influence of grader bias is excluded.

Similarly to the generation process of peer grade, the relative peer grade is generated with the given true grades of

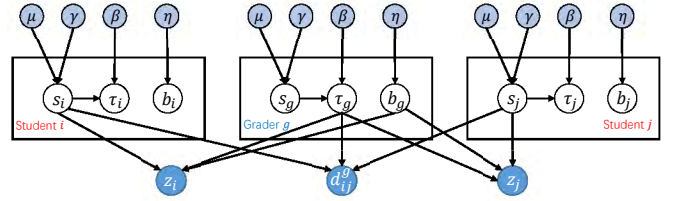


Figure 1: The plate notation of the PG_6 and PG_7 model.

Table 1: Notations

Notation	Description
S	set of all students
G	set of all graders
τ_g	reliability of grader g
b_g	bias of grader g
s_i	true grade of submission from student i
z_i^g	observed grade of submission from student i by grader g
d_{ij}^g	observed grade differences between submissions from student i and j by grader g

two submissions and the reliability of the grader. We assume the relative peer grade follows a Gaussian distribution with mean value equal to the difference of the true grades between two submissions and variance inversely proportional to the grader's reliability. Also, in the PG_7 model, λ is used to specify the scale of the variance.

Figure 1 shows the graphical structure of the PG_6 and PG_7 models. The box in the middle indicate a grader g and the first and last box indicate student i and j whose submissions are graded by grader g . Table 1 summarizes the notations of variables.

In the PG_6 model and PG_7 model, the grader's reliability τ_g and bias b_g and the submission's true grade s_i are the latent variables that need to be estimated. However, these latent variables are related to each. To estimate the values of these latent variables, Gibbs sampling is applied in this work to draw samples of a latent variable from an approximated posterior distribution. After enough iterations, we discard the first few burn-in iterations and we use the mean value of sampled s_i as the final estimate of the true score of submission i . For s_i in PG_6 and τ_g in PG_7 , we cannot find a closed form of the posterior distribution, so we use a discrete approximation to get the approximate posterior distribution of these two variables. Next we will describe the details of generation process and the inference of the PG_6 and PG_7 model separately.

4.1 The PG_6 Model

The generative process of the PG_6 model is as follows:

- For each submission submitted by student i
 - Draw true grade $s_i \sim \mathcal{N}(\mu, \frac{1}{\gamma})$
- For each grader g
 - Draw bias $b_g \sim \mathcal{N}(0, \frac{1}{\eta})$
 - Draw reliability $\tau_g \sim \Gamma(s_g, \beta)$
- For each peer grade z_i^g submitted by grader i graded by grader g
 - Draw peer grade $z_i^g \sim \mathcal{N}(s_i + b_g, \frac{1}{\tau_g})$

- For each relative peer grade d_{ij}^g between submissions submitted by student i and j graded by grader g
 - Draw relative peer grade $d_{ij}^g \sim \mathcal{N}(s_i - s_j, \frac{2}{\tau_g})$

In the PG_6 model, the posterior distribution of the true score of submission s_i does not have a closed form. To have an approximate distribution of this latent variable, in this paper, we discretized the true score of submission s_i from 0 to 15 (the full mark of the assignment) with an interval of 0.1. The variables are updated according to Eq. 1.

$$\begin{aligned} b &\sim \mathcal{N}\left(\frac{\sum_{i \in S_g} \tau_g (z_i^g - s_i)}{\eta + |S_g| \tau_g}, \frac{1}{\eta + |S_g| \tau_g}\right) \\ \tau &\sim \Gamma(s_g + \frac{|S_g|^2}{2}, \beta + \frac{\sum_{i \in S_g} (z_i^g - s_i - b_g)^2 + \sum_{i,j \in S_g} \frac{1}{2} (d_{ij}^g - (s_i - s_j))^2}{2}) \\ s &\propto \frac{\beta^{s_i} \tau_i^{s_i-1}}{\Gamma(s_i)} \times \exp\left(\frac{-R}{2} (s_i - \frac{Y}{R})^2\right) \end{aligned} \quad (1)$$

where $R = \gamma + \sum_{g \in G_i} (\frac{1}{2} \tau_g (|S_g| + 1))$, and

$$Y = \mu\gamma + \tau_g (\sum_{g \in G_i} (z_i^g - b_g) + \sum_{g \in G_i} \sum_{j \in S_g} \frac{(d_{ij}^g + s_j)}{2}).$$

4.2 The PG_7 Model

The difference between PG_7 model and PG_6 model lies in the grader reliability generation: PG_7 adopts Gamma distribution while PG_6 adopts Gaussian distribution. The generative process of the PG_7 model is as follows:

- For each submission submitted by student i
 - Draw true grade $s_i \sim \mathcal{N}(\mu, \frac{1}{\gamma})$
- For each grader g
 - Draw bias $b_g \sim \mathcal{N}(0, \frac{1}{\eta})$
 - Draw reliability $\tau_g \sim \mathcal{N}(s_g, \beta)$
- For each peer grade z_i^g submitted by grader i graded by grader g
 - Draw peer grade $z_i^g \sim \mathcal{N}(s_i + b_g, \frac{\lambda}{\tau_g})$
- For each relative peer grade d_{ij}^g between submissions submitted by student i and j graded by grader g
 - Draw relative peer grade $d_{ij}^g \sim \mathcal{N}(s_i - s_j, \frac{2\lambda}{\tau_g})$

In this model, the posterior distribution of the reliability of a grader τ_g does not have a closed form neither and we apply discrete approximation to approximate the posterior distribution of grader's reliability from 0 to 15 with an interval of 0.1. The variables are updated according to Eq. 2.

$$\begin{aligned} b &\sim \mathcal{N}\left(\frac{\sum_{i \in S_g} \frac{\tau_g}{\lambda} (z_i^g - s_i)}{\eta + |S_g| \frac{\tau_g}{\lambda}}, \frac{1}{\eta + |S_g| \frac{\tau_g}{\lambda}}\right) \\ \tau &\propto \tau_g^{\frac{|S_g|^2}{2}} \times \exp\left(\frac{-\beta}{2} [\tau_g - (s_g - \frac{\sum_{i \in S_g} (z_i^g - s_i - b_g)^2}{2\lambda\beta} - \frac{\sum_{i,j \in S_g} (d_{ij}^g - s_i + s_j)^2}{4\lambda\beta})]^2\right) \\ s &\sim \mathcal{N}\left(\frac{Y}{R}, \frac{1}{R}\right) \end{aligned} \quad (2)$$

where $R = \gamma + \beta + \sum_{g \in G_i} \frac{\tau_g}{\lambda} + \sum_{g \in G_i} \frac{\tau_g * (|S_g| - 1)}{2\lambda}$, and

$$Y = \gamma\mu + \beta\tau_i + \frac{\tau_g}{\lambda} (\sum_{g \in G_i} (z_i^g - b_g) + \frac{\sum_{g \in G_i} \sum_{j \in S_g} (d_{ij}^g + s_j)}{2}).$$

Table 2: Dataset Statistics

	Question1	Question2	Question3
# of graders	100	237	105
# of submissions	126	288	141
# of peer grades	493	1121	516
# of instructor grades	114	257	123
full grades	15	15	15
observed mean	6.8	6.7	6.2
observed variance	0.11	0.12	0.14

5. EXPERIMENTAL RESULTS

We perform experiments on a real-world dataset with three questions to evaluate the performance of the proposed models, and we show the results in this section.

5.1 Dataset

The real dataset including peer grades for three questions was collected from a course named "Immortal Arts: Approaching the masters and classics" on the XuetangX platform¹. For each question, students are asked to write an essay between 100 and 250 words. The peer graders for each submission are automatically assigned by the platform and the grading process is double-blind. After receiving the peer grades, the platform uses the median of peer grades as the final grades for submissions. The grades assigned by TAs are also available in this dataset, which we use as ground truth (true score) in evaluation. The overall statistics of this dataset is shown in Table 2.

5.2 Baselines

In order to evaluate the effectiveness of the proposed models, we compare them with 6 baselines, which are discussed as follows. including the median of peer grades, the mean of peer grades, the PG_1 model and the PG_3 mode in [7] and the PG_4 model and PG_5 model in [6].

- Median: This approach takes the media of peer grades as the final grade. This is the most frequently used method to aggregate peer grades in MOOC platforms such as Coursera² and XuetangX platform.
- Mean: This approach simply assigns the mean value of peer grades as the final grade to a submission. In some cases, using the mean value of peer grades as the final peer grades may achieve good performance according to [9].
- PG_1 : This is the first probabilistic model for cardinal peer grading estimation that considers the reliability and bias of graders [7].
- PG_3 : This is a probabilistic model that links the grader's reliability with the grader's own grade. This model assumes that the variance of distribution for the peer grades is inversely proportional to a linear function of the grader's grade [7].
- PG_4 : This is a probabilistic model assuming that a grader's reliability follows a Gamma distribution with the shape parameter equal to the grader's own grade. The PG_6 model is an extension of this model [6].

¹www.xuetangx.com

²www.coursera.org

Table 3: Experimental Results

	Question 1		Question 2		Question 3	
	Mean	Std	Mean	Std	Mean	Std
Mean	1.80		2.29		2.06	
Median	2.19		2.57		2.29	
PG_1	1.97	0.02	2.34	0.01	2.21	0.02
PG_3	1.69	0.07	2.85	0.01	1.92	0.01
PG_4	2.54	0.02	2.94	0.02	3.07	0.02
PG_6	1.31	0.01	1.44	0.01	1.38	0.02
PG_5	1.52	0.04	1.80	0.01	1.74	0.02
PG_7	1.24	0.02	1.45	0.01	1.31	0.01

- PG_5 : This is a probabilistic model assuming that a grader’s reliability follows a Gaussian distribution with the mean equal to the grader’s own grade. The PG_7 model is an extension of this model.

5.3 Experimental Settings

As described before, many hyper-parameters are used in the proposed models and baselines, and it is important to set reasonable values for these hyper-parameters. In this section, we describe how to set the values of these hyper-parameters in our experiment.

Since the proposed models are the extensions of the PG_4 and PG_5 model in [6], to evaluate the effect of leveraging relative grades, we set the same values for the shared hyper-parameters in the proposed models and the PG_4 and PG_5 models. We use the mean and variance of the peer grades as the mean (μ) and variance ($\frac{1}{\gamma}$) of the prior distribution of the true grade (s_i). As claimed in [6], the β in the PG_4 model which decides the rate of the Gamma distribution for the grader’s reliability and the λ in the PG_5 model which determines the variance of the Gaussian distribution for peer grades are the most important hyper-parameters. These parameters have a significant influence on the performance of these two models while other hyper-parameters influence the performance slightly if set in a reasonable range. Thus we mainly tune β in the PG_4 and PG_6 model and λ in the PG_5 and PG_7 model. We search for these two hyper-parameters in the range of [50, 300] with the interval of 50 to get the best performance. We set η to 0.1 in our experiment, and in the PG_5 and PG_7 model, β is set to 0.1. For each latent variable, we sample it for 300 iterations and the first 60 iterations are the burn-in iterations that will be discarded. The average results over 10 runs with the hyper-parameter settings described above are reported.

5.4 Real Dataset Performance

We use Root-Mean-Square-Error (RMSE) to evaluate the performance of the proposed models and baselines on the datasets. The experimental results are shown in Table 3. From Table 3, we can find that on all these three questions, the PG_6 and PG_7 models outperform other baselines. The RMSE of the PG_6 and the PG_7 models which incorporate the relative observed grades to capture the dependency between true grades of submissions has dropped compared with that of the PG_4 and PG_5 models. The results demonstrate the effectiveness of incorporating relative peer grades in cardinal peer grade estimation.

To better illustrate the performance of the PG_6 and PG_7 models, we further compare the estimated grades with the

ground truth on individual submissions in Figure 2. The submissions are sorted with an increasing order of the ground truth. Then we plot the estimated grades from *Mean* (the best naive method), PG_5 (the best baseline), and the proposed PG_7 model which has the best performance. We can find that the estimated grades by all three models show an increasing trend, but *Mean* shows a strong negative bias in the peer grades: the peer grades are consistently lower than the ground truth grade. Therefore, it is important to model the bias in graders to improve the aggregation results. PG_5 and PG_7 both show positive bias compared with the ground truth, but PG_5 ’s bias is a bit higher. The comparison between PG_5 and PG_7 illustrates that the relative grades can also help estimate the bias more accurately. It may imply that although graders cannot give accurate absolute grades, they can assign accurate relative grades.

We further compare the experimental bias estimated by the proposed models with the real bias. The experimental bias is defined as the average difference between the peer grades assigned by a grader and the estimated true grades. The real bias is defined as the average difference between the peer grades assigned by a grader and the ground truth. For example, a grader g grades two submissions from student i and j , the experimental bias of this grader is $\frac{(z_i^g - s_i) + (z_j^g - s_j)}{2}$

and the real bias is $\frac{(z_i^g - s_i^*) + (z_j^g - s_j^*)}{2}$, where s_i and s_j are the estimated grades, s_i^* and s_j^* are the groundtruth grades for submission i and j . The results are illustrated in Figure 3, where x-axis denotes the real bias and y-axis denotes the experimental bias. We can see that most graders are harsh graders whose real biases are less than 0. The diagonal means that the estimated bias is the same as the real bias. The closer to the diagonal, the more accurate the bias estimation is. We can observe that our estimated bias is close to the real bias. With better bias estimation, the proposed models achieve more accurate cardinal estimation. This result again indicates the informativeness of relative grades in estimating final grades.

5.5 Sensitivity of Hyper-parameters

To show how the value of hyper-parameter β in the PG_6 model and the hyper-parameter λ in the PG_7 model will influence the performance, we conduct experiments using different values of these two hyper-parameters with all other hyper-parameter fixed. In the experiment to test the sensitivity of the models, the settings for other fixed hyper-parameters are the same as described above and the β in the PG_6 model and the λ in the PG_7 model are set from 50 to 300 with an interval of 50. The results in Figure 4 show that in a reasonable range these two models are robust to the value of the parameter and achieve acceptable performance.

6. CONCLUSIONS AND FUTURE WORK

With the popularity of the MOOCs, peer assessment has become an effective paradigm for large-scale grading. The aggregation of peer grades is a challenging problem due to the various levels of bias and reliability among graders that are unknown. Existing work contributes to the development of effective peer grading aggregation methods by modeling grader bias and reliability, but they ignore an important aspect in peer grading aggregation, which is the dependency relation among grades. In these models, the relative grades are not considered and the true grades of submission are

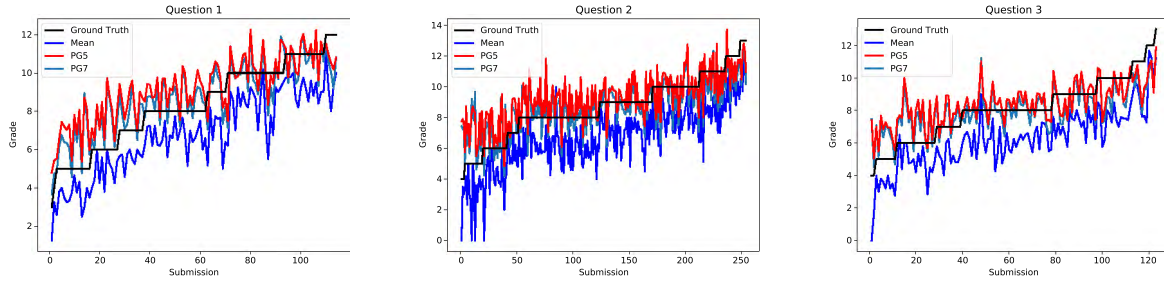


Figure 2: The estimated grades of three questions using mean, the PG_6 and PG_7 model and ground truth. The submissions are sorted by their ground truth.

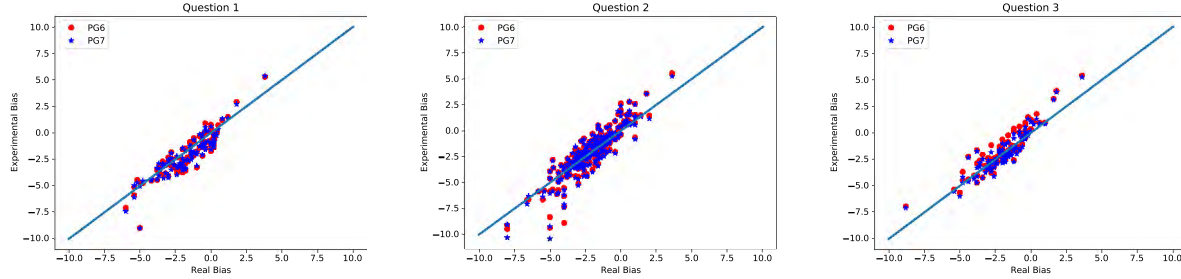


Figure 3: The comparison of experimental bias with real bias

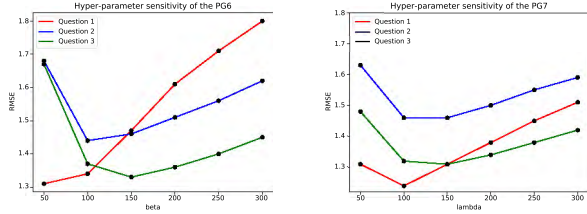


Figure 4: Hyper-parameter sensitivity of the PG_6 and PG_7 model

modeled independently. Modeling the dependencies among the true grades of different submissions can help improve the robustness of the aggregated grade estimation. In this paper, we propose two novel models that leverage relative grades to achieve improved estimation of final grades. In the proposed probabilistic models, we capture the distributions of true scores based on graders' bias and reliability degrees as well as their own submission scores which represents their knowledge about the question. In addition, the proposed models couple the true scores of different submissions via their differences. Effective inference algorithms are proposed to infer both model parameters and final scores. Experimental results demonstrate that the proposed models improve the accuracy of cardinal peer grading estimation. It can also be observed that the relative peer grades among submissions indeed contribute to the improvement in the accuracy of cardinal peer grading estimation.

In the future, we will investigate how to better model the ability of graders reflecting both reliability and bias of graders and how to cluster the graders and submissions into different groups to improve the peer assessment.

7. ACKNOWLEDGEMENTS

This work is sponsored by NSF IIS-1553411. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not neces-

sarily reflect the views of the National Science Foundation.

8. REFERENCES

- [1] A factorization approach to evaluate open-response assignments in moocs using preference learning on peer assessments. *Knowledge-Based Systems*, 85:322 – 328, 2015.
- [2] H. P. Chan and I. King. Leveraging social connections to improve peer assessment in moocs. In *Proceedings of World Wide Web Companion*, pages 341–349, 2017.
- [3] L. de Alfaro and M. Shavlovsky. Crowdgrader: A tool for crowdsourcing the evaluation of homework assignments. In *Proceedings of SIGCSE*, 2014.
- [4] J. Hamer, K. T. K. Ma, and H. H. F. Kwong. A method of automatic grade calibration in peer assessment. In *Proceedings of Computing Education*, ACE '05, 2005.
- [5] C. Kulkarni, K. P. Wei, H. Le, D. Chia, K. Papadopoulos, J. Cheng, D. Koller, and S. R. Klemmer. Peer and self assessment in massive online classes. *ACM Trans. Comput.-Hum. Interact.*, 20(6):33:1–33:31, Dec. 2013.
- [6] F. Mi and D.-Y. Yeung. Probabilistic graphical models for boosting cardinal and ordinal peer grading in moocs. In *Proceedings of AAAI*, 2015.
- [7] C. Piech, J. Huang, Z. Chen, C. B. Do, A. Y. Ng, and D. Koller. Tuned models of peer assessment in moocs. *CoRR*, abs/1307.2579, 2013.
- [8] K. Raman and T. Joachims. Methods for ordinal peer grading. In *Proceedings of SIGKDD*, pages 1037–1046, 2014.
- [9] M. S. Sajjadi, M. Alamgir, and U. von Luxburg. Peer grading in a course on algorithms and data structures: Machine learning algorithms do not improve over simple baselines. In *Proceedings of Learning @ Scale*, pages 369–378, 2016.
- [10] N. B. Shah, J. K. Bradley, A. Parekh, and K. Ramchandran. A case for ordinal peer-evaluation in moocs. 2013.
- [11] T. Walsh. The peerrank method for peer assessment. *arXiv preprint arXiv:1405.7192*, 2014.
- [12] A. E. Waters, D. Tinapple, and R. G. Baraniuk. Bayesrank: A bayesian approach to ranked peer grading. In *Proceedings of Learning @ Scale*, 2015.

Toward Near Zero-Parameter Prediction Using a Computational Model of Student Learning

Daniel Weitekamp III
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
weitekamp@cmu.edu

Erik Harpstead
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
harpstead@cmu.edu

Christopher J. MacLellan
Soar Technology, Inc.
3600 Green Court, Suite 600
Ann Arbor, MI 48105
chris.maclellan@soartech.com

Napol Rachatasumrit
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
napol@cmu.edu

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
koedinger@cmu.edu

ABSTRACT

Computational models of learning can be powerful tools to test educational technologies, automate the authoring of instructional software, and advance theories of learning. These mechanistic models of learning, which instantiate computational theories of the learning process, are capable of making predictions about learners' performance in instructional technologies given only the technology itself without fitting any parameters to existing learners' data. While these so call "zero-parameter" models have been successful in modeling student learning in intelligent tutoring systems they still show systematic deviation from human learning performance. One deviation stems from the computational models' lack of prior knowledge—all models start off as a blank slate—leading to substantial differences in performance at the first practice opportunity. In this paper, we explore three different strategies for accounting for prior knowledge within computational models of learning and the effect of these strategies on the predictive accuracy of these models.

1. INTRODUCTION

A computational theory approach to modeling psychological phenomenon consist of building simulations of cognitive processes and testing their ability to emulate and explain human behavior. Within educational data mining there have been several attempts to model student learning processes using a computational theory approach. For example SimStudent [1] is a computational approach which simulates human students. The Apprentice Learner (AL) Architecture [2] is a modular framework for creating computational agents such as SimStudent that learn to solve problems in intelligent tutoring systems (ITSs) as a human student would. These computational agents mimic the inductive learning process

undergone by students in response to examples and correctness feedback given in a particular domain.

These approaches afford several different use cases relevant to the domain of EDM. First, as cognitive models of human learning, these computational agent models can be used to test theories of human learning. The AL framework uniquely serves this role since it is designed so that its components can be interchanged, to enable the instantiation of different theories of learning that can be tested against human behavior.

Second, insofar as as computational agents constitute high fidelity models of human learning as simulated students, they can be used as cognitive crash dummies for instructional design prior to the longer process of classroom trials and A/B studies. For example, experiments with SimStudent and agents built with AL showed that interleaving fraction addition and fraction multiplication problems would lead to more efficient learning by opportunity in an ITS [14]. This result was later corroborated by human trials [15].

Finally, simulated students can be used as efficient authoring tools. Simulated students can form generalized production rules from examples and correctness feedback meaning that they can be trained in a manner similar to example tracing [3] to build expert models for ITSs. Expert models trained with SimStudent, for example, have similar expressivity and accuracy as hand-written production rule models, but can be built in significantly less time and without any programming knowledge required by the author [12, 11, 16].

Computational theory approaches of learning such as SimStudent and the simulated students built with the AL framework are at their core zero-parameter models of human learning. These approaches are distinct from performance pattern models such as the Additive Factors Model (AFM) [4, 5] or Bayesian Knowledge Tracing (BKT) [6] since they attempt to predict the presence of a pattern in behavior (e.g., learning a new skill or misconception) given only the task environment and a few underlying assumptions about the learning process rather than fitting to data from individual students. These computational theory approaches to human

Daniel Weitekamp, Erik Harpstead, Napol Rachatasumrit, Christopher Maclellan and Kenneth R. Koedinger "Toward Near Zero-Parameter Prediction Using a Computational Model of Student Learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 456 - 461

modeling attempt to make predictions of human behavior without knowing anything about the humans that they are modeling, putting the burden of the modeling process on deliberate and explainable algorithmic design choices. While there are added challenges to designing computational theory approaches that accurately reflect human behavior, they have the advantage of fine-grained explainability. Since a computational theory approach actually performs the tasks given to the students that it attempts to model, the fidelity of the model to its human counterparts can be evaluated in a more incisive manner, considering not just the performance of the model by opportunity, but the types of errors it makes and strategies it employs as well.

While computational theory models have demonstrated some success in the past they still possess several systematic issues in their performance. When applied to the task of modeling learning many computational theory models manifest some version of a cold start problem. Namely, as these models, by definition, are not fit to information from individual students, they often lack the ability to account for individual differences in prior knowledge or experience that students bring to bear. Performance pattern models are free to deal with this problem in a number of ways. For example, AFM and its variants often fit some kind of student intercept to allow for variation among students' initial ability [4, 5]. Several strategies for estimating Bayesian priors in BKT have been proposed in the EDM literature [6, 7]. In a performance pattern context estimating prior knowledge amounts to estimating some parameter between 0 and 1. Computational theory models, on the other hand, require a fully specified mechanistic skill to be initialized and refined in a learning process that directly models the process of a human trying to master skills in an intelligent tutoring system.

In this paper we explore several strategies for accounting for prior knowledge in the AL architecture and compare each strategy's ability to generate learner performance similar to that of students. We close with a discussion of these strategies and discuss the limitations of our current approaches as well as directions for future work.

2. OUR EXPERIMENT

2.1 The Apprentice Learner Architecture

The Apprentice Learner (AL) Architecture is a modular framework for modeling the human learning process from demonstrations and feedback [2]. The purpose of the architecture is to serve as a test-bed for computational theory approaches to student modeling. Agents created in the AL framework induce production rules from training that can be provided either interactively by a human or by working with an existing ITS. In our experiments we use the latter approach to train a set of simulated students on an intelligent tutoring system with the same order of problems as each student from a human dataset.

AL agents receive two types of feedback from ITSs, examples and correctness feedback. When an AL agent has a learned production rule that can fire it will try the resulting action and get correctness feedback from the ITS, otherwise the agent will request a hint (i.e., an example) from the ITS. Through positive examples from hint requests and positive feedback, and negative examples from negative feed-

back, AL refines its understanding of the domain and learns to correctly solve problems in the ITS.

To induce a production rule model suitable for solving problems, AL employs a four mechanism learning process consisting of *when-learning*, *where-learning*, *how-learning*, and *which-learning*. The *when-* and *where-learning* mechanisms determine the context in which a production rule should fire, the *how-learning* mechanism searches for chains of overly general operators that explain tutor demonstrations to determine what a production should do when it fires, and the *which-learning* determines which production rule should fire if multiple are applicable.

The operators that *how-learning* employs are "overly general", in the sense that they are applicable in a wide range of domains, but not specific enough to solve problems in any particular domain. Overly general operators do not constitute production rules since they are not programmed with any sense of when they should fire. In our experiments we use one perceptual operator 'equals' which adds to the problem state the fact that two values are equal, and four procedural operators 'add', 'subtract', 'multiply' and 'divide'. The conditions in which these operators should be utilized, the particular interface elements from which they derive their numerical inputs, and the interface elements that will receive their outputs are learned by the *when-* and *where-learning* mechanisms respectively to create full production rules.

The AL agents in our experiments employ Trestle [13], an incremental hierarchical categorization algorithm for *when-learning*. For *where-learning*, since we use only static interfaces in our experiments, we employed a 'most specific' strategy, which uses the inputs and outputs present in positive training examples without trying to generalize from those examples to unseen cases. We allow our *how-learning* mechanism to only use single operation explanations (i.e. just multiply or add two numbers, not three numbers or combinations of operators). The *which-learning* mechanism chooses the production rule which has been employed successfully with the highest frequency so far given the current state representation.

2.2 Data Source

To evaluate different methods of pretraining AL agents we used student performance data from an ITS for fraction arithmetic [15]. The dataset consisted of the work of 117 students on three different types of fraction arithmetic problems: 1) adding fractions with the same denominator, 2) adding fractions with different denominators, and 3) multiplying fractions. In the interface students are first asked if they need to convert the two fractions, which is false in the addition-same and multiplication cases and true in the addition-different case. If the fractions need to be converted then the students must find the common denominator between them and write in the converted fractions before adding them. This dataset is available on DataShop [9]¹.

¹<https://pslcdatashop.web.cmu.edu/Project?id=243>

2.3 Training Strategies

We explored three different strategies for accounting for prior knowledge in AL Agents. For each of these strategies an agent was paired with a human student and provided some form of pretraining based on data collected about that student prior to working through that student's problem sequence in the data. In our experiments we attempted three different strategies to account for prior knowledge estimated fraction prior experience, estimated whole number prior experience, and demonstrated pretest. Additionally, we ran a no pretraining control condition which begins from a conventional cold start.

2.3.1 Estimating Prior Experience

In the estimated fraction and estimated whole number prior experience strategies agents worked through a number of pretraining problems based on estimates of each student's prior opportunities to practice problems in each of the three types of fraction arithmetic problem. We estimated the number of prior opportunities (P_{ik}) that a student i had on each knowledge component (KC) k [10] in the fraction arithmetic tutor. Taking the AFM mixed model regression equation:

$$\log\left(\frac{p_{ij}}{1-p_{ij}}\right) = \theta_i + \sum_k (q_{jk}\beta_k + q_{jk}\gamma_k T_{ik}), \theta_i \sim \mathcal{N}(0, \sigma^2)$$

Consider an imaginary tabula-rasa student with no knowledge of anything at all with student intercept $\theta_{-\infty}$, and a student with student-intercept θ . We can find the number of prior opportunities P_{ik} such that the tabula-rasa student has the same log odds of answering a question in KC k as student i :

$$\begin{aligned} \theta_{-\infty} + \sum_k (q_{jk}\beta_k) + \sum_k (q_{jk}\gamma_k P_{ik}) &= \theta_i + \sum_k (q_{jk}\beta_k) \\ \Rightarrow P_{ik} &= \frac{\theta_i - \theta_{-\infty}}{\gamma_k} \end{aligned}$$

An issue with this formulation is that a true tabula-rasa student would have $\theta_{-\infty}$ equal to $-\infty$, so we introduce an approximation for $\theta_{-\infty}$ in order to get reasonable values for P_{ik} . In our experiments we choose $\theta_{-\infty} = -2$ since this is the student intercept at which a student practicing a single KC step with a KC intercept of zero would have about a 10% probability of getting the step correct. Although this an arbitrary choice, we believe it is a reasonable one, much in the same way that it is reasonable to choose a 90% chance of correct behavior as a mastery threshold in a BKT model.

There are three distinct types of fraction arithmetic problems, which each have their own sets of KCs. In the case of fraction addition the two fractions can either have the same denominator, in which case the student need only add the numerators, or the denominators could be different, requiring the student to convert the fractions before adding. In the third case, fraction multiplication, the KCs are distinct from the addition cases. Since in practice fraction arithmetic problems are not presented with any of their KCs in isolation we estimated the number of problems of each type to give as prior training to our learning agents as the minimum P_{ik} among the problems of a given type. For the

estimated fraction prior experience case we pretrain on a number of random problems from each type according to these estimates. In the estimated whole number prior experience case we pretrain as many random whole number addition problems as there are estimated same denominator problems and as many random whole number multiplication problems as estimated multiplication problems.

In the estimated fraction condition, randomly generated problems were restricted to have denominators between 2 and 12 with numerators less than each fraction's denominator (i.e., no improper fractions). In the estimated whole number condition, each agent was given randomly generated whole number arithmetic problems prior to beginning the core tutoring sequence. These whole number arithmetic tutors were restricted to numbers between 1 and 12.

2.3.2 Demonstrating Prior Answers

In the demonstrated pretest case we pretrain the AL agents by providing them with demonstrations of the exact answers that students gave in a pretest evaluation of the original study. In order to model both the knowledge and misconceptions of the student, these answers are given to AL as positive examples regardless of whether or not each of the students' answers were correct. The goal here is that the pretest encapsulates a picture of each students' prior conceptions that may contain more information than a fit parameter. The pretest is a snapshot of students behavior under a particular set of problems, a sample which we use to infuse an associated AL agent with the same knowledge and misconceptions as the original student.

One limitation of our pretest demonstration approach is that the available pretest data only contained the given fraction problems and the students' final answers. Thus, the demonstrations that we provided to the agents appeared as single steps even though the human students may have done multiple mental calculations to arrive at their answer. This issue is likely to be particularly salient on fraction addition problems with different denominators as the common denominator process would not be apparent.

3. RESULTS

In Figure 1 we see that the estimated fraction condition gets closest to the human first opportunity performance, followed by the demonstrated pretest condition. The estimated whole number condition behaves almost equivalent to the control case with a 100% first opportunity error rate.

To test the fit of each strategy to the human data we calculate the residuals between the human learning curves and the learning curves generated from the AL agents run with each pretraining strategy (Figure 2). Table 1 shows several statistics of the fit of each strategy to the human data. The "Accuracy" column shows the mean accuracy between the correctness (0 for incorrect and 1 for correct) of each human and that human's AL agent counterpart over all students and opportunities. The "First Opp. Accuracy" column shows this same statistic, but only for the first opportunity. In both cases the estimated fraction strategy shows the best fit to the human data.

In addition to testing the predictive accuracy of the models,

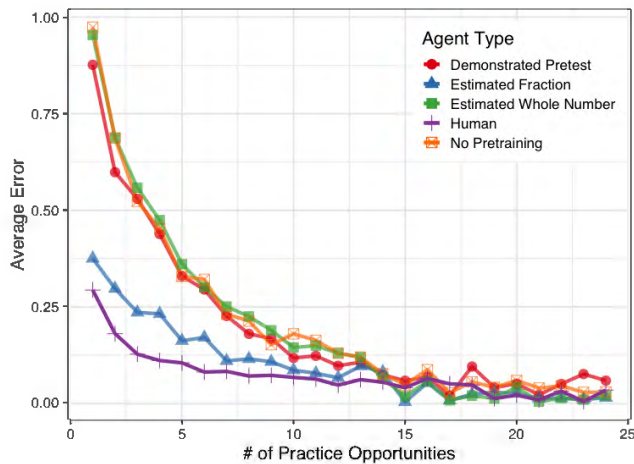


Figure 1: Learning curves aggregated over all knowledge components for each pertaining strategy and the human learning curves.

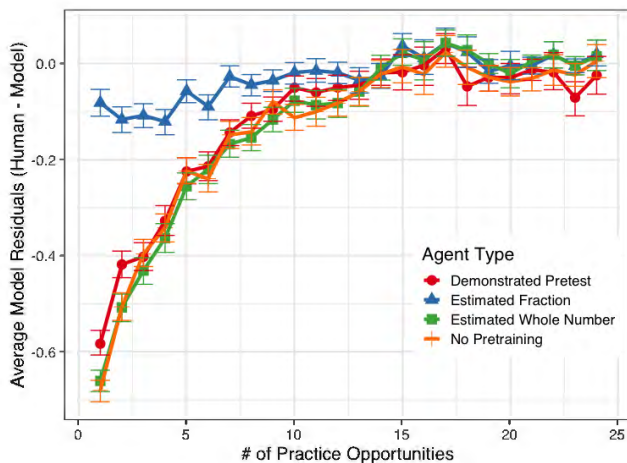


Figure 2: Learning curve residuals (human minus agent) for each strategy across all knowledge components. Error bars are standard errors.

we next looked at the explanatory power of the model— by looking at its ability to account for variation in the human error rates over the course of problem solving. As a coarse measure of explanatory power, we first conducted a χ^2 test of independence between each model correctness and the human correctness. This test confirmed that there is a significant relationship between each models’ predictions and the human correctness (independence hypothesis rejected for all models, $p < 0.001$). Next, we looked to more finely evaluate how well each model explains variation in the human data. To do this we used a mixed-effects regression analysis [16]. For this analysis, we used a mixed-effects model with fixed effects for each model prediction as well as the practice opportunity counts. The model also included random effects for the intercept and slopes of each knowledge components as well as intercepts for each student.² This model,

²The mixed-effect regression model used in R: $\text{Human.Correctness} \sim \text{Agent.Correctness} + \text{Opportunity} + (1$

which is effectively the Additive Factors Model [4, 5] with an additional term for the model predictions, serves a role similar to a repeated measures ANOVA analysis by accounting for general effects of within student and skill (knowledge component) performance as well as the effect of repeated practice. The model enabled us to evaluate how well each computational models’ predictions improved the overall regression model fit over the baseline AFM model. By applying model fit statistics, such as AIC, to these mixed-effects models, we could evaluate which model better accounted for variation in the human behavior above general practice effects and general correlations of behavior within students and skills. The baseline AFM model fit to this data had an AIC score of 9558.7. The “AIC” column in Table 1 shows the AIC scores for the mixed-effects model with the added fixed effects for each respective computational model. Note, the numbers of free parameters and data points between each of these models is equivalent thus we did not consider other fit statistics such as BIC or HQC as they would have resulted in an identical ordering.

These results show that all models provide some explanatory power over the baseline AFM model. However, the no pretraining control model appears to provide the most explanatory power. Although this result seems counter-intuitive from the graphs shown in Figure 1, there is at least one possible explanation. Mainly, that the estimated fraction model better fits the average of all students whereas the no pretraining model better fits individual students—specifically students with high initial error. The estimated fraction model likely better fits the students that perform better initially; however these students have less variation in their behavior that the model might account for and this variation is already accounted for as a general within-student effect of the regression evaluation. In contrast, the no pretraining control and demonstrated pretest models better fit students with lower performance at the beginning, who also have more variation to account for.

Finally, we analyzed the residuals shown in Figure 2 using a linear regression analysis. This analysis fit a line to each residual curve to get an estimate of the intercept and slope of these curves. Table 2 shows the slopes and intercepts of these linear models with their accompanying 95% confidence regions. This analysis shows there is a significant intercept and slope for each of the models (all intercepts and slopes are non-zero, $p < 0.001$). However, the estimated fraction model has the intercepts and slopes that are closest to zero, suggesting that it is a better fit of the overall error rates and error rates by opportunity.

+ Opportunity | KnowledgeComponent) + (1 | Student.Id).

Table 1: Fit statistics to human data by strategy.

Strategy	Acc.	First Opp. Acc.	AIC
No Pretraining	0.684	0.316	9541.3
Est. Fractions	0.805	0.643	9558.1
Demo Pretest	0.701	0.389	9548.2
Est. Whole Num	0.681	0.326	9556.4

4. DISCUSSION

Our current experiments in accounting for prior knowledge in AL agents have mixed results, but yield fruitful directions for future work. Our most promising method, the estimated fraction prior experience case, reduces the average first opportunity error rate across all knowledge components down to about 40%. The human students by contrast begin at about a 30% error rate. The demonstrated pretest case yielded little improvement over the control. The only slight first opportunity improvement in the demonstrated pretest case is likely due to the often erroneous answers that the students produced in the pretest, which we presented to our AL agents as the ground truth in this condition. Our original hypothesis was that these erroneous but positively labelled examples would account for any existing misconceptions in the human students. However, these erroneous examples may have gone too far toward confusing the AL agents. An alternate explanation is that the lack of negative examples in this case hinders the AL agents' capacity to effectively delineate between their learned production rules. Finally, since the estimated whole number learning curves turned out to behave equivalently to the control case both in first opportunity error rate and in over-all shape, we need to look more closely at AL's capabilities for cross-interface skill transfer in future work.

We consider these experiments to be a first exploratory pass at accounting for prior knowledge in computational modeling approaches to human learning. Among them, the estimated fraction prior experience case is decidedly the closest to human behavior. However, we certainly think we can get much closer to the human behavior going forward.

One direction for future work is to improve our estimates for the number of opportunities to pre-train our agents with to account for the prior knowledge of their human counterparts so that the intercepts of the human and AL agent learning curves match. We can approximate the average of the ground truth number of prior opportunities by calculating the number of opportunities that the control condition takes to gain parity with the human students at their first opportunity. By comparison to this ground truth average it appears that our estimates are too large in the multiplication (14.17 vs. 6) and same denominator addition problems (6.76 vs. 4) and too low in the addition different problems (6.79 vs. 9). We found that these discrepancies could not be remedied by a different choice of $\theta_{-\infty}$.

In previous evaluations of the AL framework in other domains AL agents learned more per opportunity than the human subjects [16]. We hope to explore this fact further by evaluating situations where AL agents over/under performs relative to the humans, and find ways of correcting the discrepancy.

Table 2: Linear model fit to residuals.

Strategy	Intercept	Slope
No Pretraining	-0.474 ± 0.012	0.030 ± 0.002
Est. Fractions	-0.108 ± 0.011	0.007 ± 0.002
Demo Pretest	-0.422 ± 0.012	0.026 ± 0.001
Est. Whole Num	-0.490 ± 0.012	0.031 ± 0.001

One direction for future work that will help us converge on an accurate model of a human learning, is to give AL agents the capability to make random guesses guided by statistics of human errors. This behavior might include inputting common random numbers, copying random numbers, and applying overly general operators at random. Much in the same way that we have estimated the P_{ik} from human data, we would need to estimate a distribution from which these behaviors could be drawn. Similar statistics could be estimated for inferential and procedural slip mistakes³.

The question certainly is raised: if we must rely on descriptors of human behavior derived from fitting statistical models to human performance, then to what extent are we employing a zero-parameter approach to modeling? No doubt, we are using fit parameters in our modeling approach by bootstrapping our models guided by student intercepts calculated from AFM. However, our computational modelling approach attempts to replicate human behavior instead of only fitting performance, and this process is guided a priori via computational theories of student learning, not by comparison to human data.

Although our nearly zero-parameter computational approach opens up many possibilities for testing theories of human learning, it comes with its own set of difficulties and limitations. In contrast to performance estimation models such as AFM and BKT which require only the logs of human performance on an ITS, our approach also requires a working ITS for the AL agent to work against. This limits the applicability of our method to older datasets for which the actual tutoring interfaces have been lost to time. Currently our system works on the newest HTML version of CTAT and has been used successfully in previous experiments on the older Java version of CTAT as well. However, in order to use different ITSs new code must be written to communicate tutor events to the AL framework's RESTful API.

While a nearly zero-parameter computational approach to human learning allows us to form and test theories of learning in a very detailed manner, it should be noted that the strength of any claim about the underlying cognitive processes of students is dependant on both the specificity of such a model in replicating human behavior and the generalizability of such a model across different domains. Here we have looked at just a single domain, fraction arithmetic, since it is amenable to all three of our proposed strategies. However, it should be noted that although validity claims are strengthened in a zero-parameter computational approach by the fact that behavior must be explained on an algorithmic level and not by fitting parameters, it is still possible that two underlying learning mechanisms yield almost indistinguishable behavior. Thus generalizability is essential to any claim about human cognitive processes. However, generalizability cannot be gained for free by the availability of more diverse data, as is often the case with deep learning models. Rather, any observed discrepancies between domains must be explicitly explained and accounted for by the investigator at an algorithmic level. This fact lends very high explainability to a zero-parameter computational approach, but nonetheless presents an added challenge.

³These statistics would be difficult to estimate convincingly with a zero-parameter approach.

5. CONCLUSION

We have tested three different strategies for accounting for prior knowledge in AL agents trained on fraction arithmetic problems. Our three strategies were 1) 'estimated fraction prior experience' where we gave the AL agents additional practice in fraction arithmetic problems before starting the core tutor problems, 2) 'demonstrated pretest' where we showed the AL agents their human counterparts' answers on pretest problems before starting the core tutor problems, and 3) 'estimated whole number prior experience' where we pretrained the AL agents with whole number arithmetic problems. Our results showed that in terms of matching human learning curves the estimated fraction case beats out all the other strategies. However the control case, which had no pretraining at all, best explains the variance in the data.

We have discussed several limitations of our nearly zero-parameter computational approach to testing theories of learning, and have offered several avenues for improvement. Concerning the limited accuracy of our current approach in estimating the number of prior opportunities necessary for an AL agent to account for the prior knowledge of its human counterpart, we have offered an avenue for further refinement. Additionally we have discussed ways that we might accounting for non-deterministic human behavior such as initial guessing. Finally, we have addressed the zero-parameter nature of our approach and considered its technical and epistemological limitations and strengths. We consider the AL framework to be a strong avenue for testing theories of learning, and hope to refine our computational approach to modeling human learning in future work.

6. ACKNOWLEDGEMENTS

The research reported here was supported in part by a training grant from the Institute of Education Sciences (R305B150008). Opinions expressed do not represent the views of the U.S. Department of Education.

7. REFERENCES

- [1] Matsuda, N., Cohen, W. W., Sewall, J., Lacerda, G., & Koedinger, K. R. (2007). Predicting students' performance with simstudent: Learning cognitive skills from observation. *Frontiers in Artificial Intelligence and Applications*, 158, 467.
- [2] MacLellan, C. J., Harpstead, E., Patel, R., & Koedinger, K. R. (2016, June). The Apprentice Learner architecture: Closing the loop between learning theory and educational data. In *EDM* (pp. 151-158).
- [3] Aleven, V., McLaren, B., Sewall, J., & Koedinger, K. R. (2009). Example-tracing tutors: A new paradigm for intelligent tutoring systems. *Int J of AI in Education*, 19, 105-154.
- [4] Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning Factors Analysis: A general method for cognitive model evaluation and improvement. In M. Ikeda, K.D. Ashley, T.- W. Chan (Eds.) *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 164-175. Berlin: Springer-Verlag.
- [5] Pavlik Jr., P.I., Cen, H., Koedinger, K.R.: Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. In: Barnes, T., Desmarais, M., Romero, C., Ventura, S. (eds.) *Proceedings of the the 2nd International Conference on Educational Data Mining*, Cordoba, Spain, pp. 121-130 (2009).
- [6] Baker, R. S., Corbett, A. T., & Aleven, V. (2008, June). More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *International conference on intelligent tutoring systems* (pp. 406-415). Springer, Berlin, Heidelberg.
- [7] Hawkins, W. J., Heffernan, N. T., & Baker, R. S. (2014, June). Learning Bayesian knowledge tracing parameters with a knowledge heuristic and empirical probabilities. In *International Conference on Intelligent Tutoring Systems* (pp. 150-155). Springer, Cham.
- [8] Aleven, V., McLaren, B. M., Sewall, J., & Koedinger, K. R. (2006, June). The cognitive tutor authoring tools (CTAT): preliminary evaluation of efficiency gains. In *International Conference on Intelligent Tutoring Systems* (pp. 61-70). Springer, Berlin, Heidelberg.
- [9] Koedinger, K. R., Baker, R. S., Cunningham, K., Skogsholm, A., Leber, B., & Stamper, J. (2010). A data repository for the EDM community: The PSLC DataShop. *Handbook of educational data mining*, 43, 43-56.
- [10] Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Science*, 36(5), 757-798. <https://doi.org/10.1111/j.1551-6709.2012.01245.x>
- [11] MacLellan, Koedinger & Matsuda (2014). Authoring Tutors with SimStudent: An Evaluation of Efficiency and Model Quality. *Proc of Intelligent Tutoring Systems*, 551-560.
- [12] Matsuda, Cohen, Koedinger (2015). Teaching the teacher. *Int J of AI in Ed*, 25, 1-34.
- [13] MacLellan, C. J., Harpstead, E., Aleven, V., & Koedinger, K. R. (2016). Trestle: a model of concept formation in structured domains. *Advances in Cognitive Systems*, 4, 131-150.
- [14] Li, N., Cohen, W. W., & Koedinger, K. R. (2012). Problem Order Implications for Learning Transfer. In *Proceedings of Intelligent Tutoring Systems*, 185-194.
- [15] Patel, R., Liu, R., & Koedinger, K. R. (2016). When to Block versus Interleave Practice? Evidence Against Teaching Fraction Addition before Fraction Multiplication. In *Proceedings of the 38th annual meeting of the cognitive science society*. Philadelphia, PA.
- [16] MacLellan, C. (2017). *Computational Models of Human Learning: Applications for Tutor Development, Behavior Prediction, and Theory Testing* (Unpublished doctoral dissertation). Carnegie Mellon University, Pittsburgh, Pennsylvania

Do Learners Know What's Good for Them? Crowdsourcing Subjective Ratings of OERs to Predict Learning Gains

Jacob Whitehill
Worcester Polytechnic
Institute, USA
jrwhitehill@wpi.edu

Cecilia Aguerrebere
Fundación Ceibal, Uruguay
caguerrebere@ceibal.edu.uy

Benjamin Hylak
Worcester Polytechnic
Institute, USA
bhylak@wpi.edu

ABSTRACT

We explored¹ how learners' *subjective ratings* of open educational resources (OERs) in terms of how much they find them "helpful" can predict the actual *learning gains* associated with those resources as measured with pre- and post-tests. To this end, we developed a probabilistic model called GRAM (Gaussian Rating Aggregation Model) that combines subjective ratings from multiple learners into an aggregate quality score of each resource. Based on an experiment we conducted on Mechanical Turk ($n = 304$ participants with $m = 17$ math tutorial videos as resources), we found that aggregated subjective ratings are highly (and stat. sig.) predictive of the resources' average learning gains, with Pearson correlation of 0.78. Moreover, when predicting average learning gains of *new* learners, subjective scores were still predictive (Pearson correlation of 0.49) and attained higher prediction accuracy than a model that directly uses pre- and post-test data to estimate learning gains for each resource. These results have potential implications for large-scale learning platforms (e.g., MOOCs, Khan Academy) that assign resources (tutorials, explanations, hints, etc.) to learners based on the expected learning gains.

Keywords

open educational resources (OER); adaptive learning; crowdsourcing; treatment effect estimation

1. INTRODUCTION

Consider a hypothetical large-scale online learning platform in which learners engage with open educational resources (OERs) that are sampled from a vast collection. These resources could include tutorial videos, practice exercises, explanations of wrong answers, hints, etc. In order to help students learn optimally, the learning platform must decide

¹The data and source code (in R) to reproduce the results in this paper are available at <https://github.com/jwhitehill/gram>.

Jacob Whitehill, Cecilia Aguerrebere and Benjamin Hylak "Do Learners Know What's Good for Them? Crowdsourcing Subjective Ratings of OERs to Predict Learning Gains" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 462 - 467

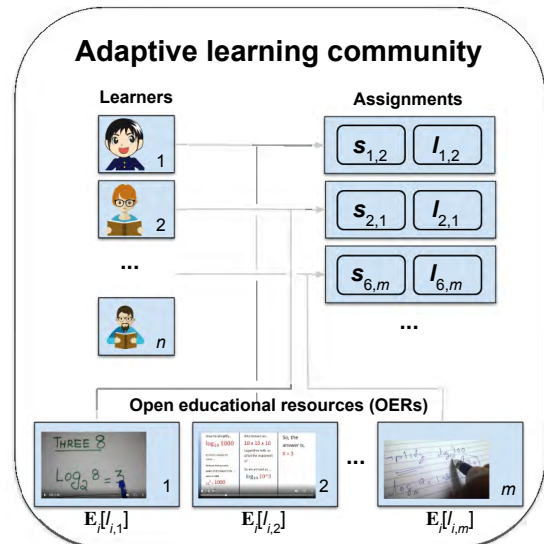


Figure 1: An adaptive learning community in which each learner i is assigned different resources over time, and the effectiveness (expected learning gains l_{ij}) of each resource j is estimated both from test scores as well as from *subjective ratings* s_{ij} given by the learners. Gray lines show hypothetical assignments of OERs to learners. $E_i[l_{ij}]$ denotes the average learning gains over all learners i who received j .

which resource is most beneficial to each learner at each moment in time, and then assign that resource to the learner (see Figure 1). Although various criteria could be used for this decision (e.g., the impact on student engagement), perhaps the most natural one is how much the student will learn – *learning gains* – from receiving the resource.

The standard way to estimate the *learning gains* l_{ij} of each resource is to give each student i who receives resource j a pre-test (before receiving it) and post-test (after receiving it) to measure how much she/he learned, i.e., the difference between pre- and post-tests. We call each (learner, resource)-pair an *assignment*. After a sufficient number of assignments, the average learning gains of each resource $E_i[l_{ij}]$ (averaged over all learners i who receive j) can be estimated. Then, using these estimates for all the resources, the most effective ones can be served to students. Unfortunately, this approach to estimating the quality of a large collection of

OERs is expensive because testing takes a long time. On the other hand, after receiving a resource j , learners may have a *subjective opinion* about how effective j was. These opinions can arguably be queried more easily and efficiently than administering tests; for example, the learner could simply select between 1 and 5 stars (à la Yelp) to express how much she/he liked it. It is even possible that subjective scores might be better than test scores in some situations. For example, even if a learner her/himself has already mastered a skill and thus has a learning gain of 0, she/he might still be able to *judge* whether a resource is useful.

When using subjective scores to predict learning gains, care must be taken: some learners may be more or less reliable in making such judgments. However, there are reasons to be optimistic: (1) As long as enough learners “vote”, then the noise of their judgments can be averaged out. (2) Using algorithms for crowdsourcing consensus (see below), the reliabilities of the learners as well as the learning gains of the resources can be estimated in an unsupervised fashion. The **chief contribution** of our work is to propose and evaluate experimentally an efficient crowdsourcing model to estimate the quality of a set of learning resources by combining multiple learners’ subjective opinions about them.

2. RELATED WORK

Students’ judgments of learning and teaching: Estimating the learning gains of an OER is related to metacognition. The ability of students to judge how well *other* people learn has been analyzed experimentally in prior works such as [12, 3]. However, we are not aware of previous research that considers this problem in the large scale of an online learning community or how to combine multiple learners’ judgments to improve accuracy. In the context of student course evaluations, there is evidence that learners may actually be poor judges of their teachers’ effectiveness [7, 4].

Adaptive online learning communities: Adaptive learning communities that decide which resources to serve to students based on up-to-date estimates have generated recent interest in the educational data mining and reinforcement learning communities. Notable works are by Rafferty, et al. [9] and Williams, et al. [17]. In these works, reinforcement learning techniques based on bandits and Thompson sampling were used both to estimate the learning gains of each resource and simultaneously to assign resources to learners. Our work is complementary: we explore how not only test score information, but subjective ratings provided by learners, could be useful in estimating the utility of each resource.

Crowdsourcing for education: In [14], Weld et al. provided an overview of how online learning creates challenges due to its large scale, but also suggests possible ways in which crowdsourcing can offer solutions to these challenges. Heffernan, et al. [6] proposes a vision of how crowdsourcing can help provide important functionality toward adaptive personalized online learning. As one specific instance of how the crowd can contribute new resources to an online learning community, Williams, et al. showed that people on Mechanical Turk can be induced to author novel and useful text-based explanations [17]. Whitehill & Seltzer [16] showed that Mechanical Turk workers can even create entire tutorial videos, at least some of which are effective at help-

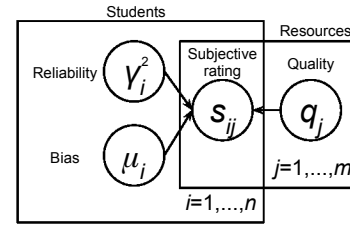


Figure 2: Gaussian Rating Aggregation Model (GRAM). Only the subjective ratings s_{ij} from student i about resource j are observed. Latent variable q_j expresses the “quality” of resource j and is used to predict the learning gains of students who receive the resource.

ing students to learn. Peer grading (e.g., Piech, et al. [8]) and peer feedback are other ways of harnessing the crowd to provide useful feedback for learners at scale.

Crowdsourcing consensus algorithms: Since Dawid and Skene’s seminal work [5] on optimal weighting of annotators’ opinions, there have been a slew (e.g., [15, 10, 11, 2, 13, 1]) of crowdsourcing models, which are suitable for different kinds of tasks (binary, multiple choice, etc.) and capture different features of the labeling task (e.g., task difficulty, biases).

3. GAUSSIAN RATING AGGREGATION MODEL (GRAM)

We model the *quality* of each open educational resource (OER) j with a real number, q_j , that can be estimated by aggregating over many (real-valued) *subjective ratings* s_{ij} from many learners i . We thus develop a Gaussian probability model of how each s_{ij} is related to each q_j as well as several parameters specific to each learner i . The model is portrayed in Figure 2: Let μ_i and γ_i^2 be the bias and reliability (variance) of learner i , respectively. Let q_j be the ground-truth quality of resource j . We posit that student i ’s label s_{ij} for resource j is a Gaussian random variable with mean $q_j + \mu_i$ and variance γ_i^2 . In other words, if the ground-truth quality is q_j , then student i adds a bias μ_i , and then adds independent 0-mean Gaussian noise with variance γ_i^2 . We can express these relationships using the conditional probability density function (PDF) $P(s_{ij} | q_j, \mu_i, \gamma_i^2) = \mathcal{N}(q_j + \mu_i, \gamma_i^2)$ where \mathcal{N} is a Gaussian with a given mean and variance.

3.1 Inference

As with many crowdsourcing consensus models, inference in the GRAM requires solving a “chicken-and-the-egg” problem: if the parameters μ_i, γ_i^2 of each learner i were known, then an optimal weighting of their votes s_{ij} could be used to estimate the quality q_j of each resource j . On the other hand, if the ground-truth quality q_j of each resource were known, then the parameters of each learner could be estimated. We solve this problem using Expectation-Maximization: in the E-Step we compute the PDF of each q_j conditional on the parameters $\{\mu_i, \gamma_i\}$. In the M-Step, we compute the expected joint log-likelihood of the $\{q_j\}$ and $\{s_{ij}\}$ w.r.t. the PDFs computed during the previous E-Step, and then maximize this expectation w.r.t. the parameters $\{\mu_i, \gamma_i\}$. Since the GRAM is Gaussian, both the E- and M-Steps can be done analytically, and thus the algorithm is very efficient.

Let the mean and variance of the prior distribution over each q_j be \bar{m} and \bar{s}^2 , respectively. Recall that the product of two Gaussian PDFs, with means m_1 and m_2 and variances s_1^2 and s_2^2 , respectively, is also Gaussian and has a mean $s^2(m_1/s_1^2 + m_2/s_2^2)$ and variance $s^2 = (1/s_1^2 + 1/s_2^2)^{-1}$.

E-Step:

$$\begin{aligned}
\tilde{P}(q_j) &\doteq P(q_j \mid \{s_{ij}\}, \{\mu_i\}, \{\gamma_i^2\}) \\
&\propto P(q_j \mid \{\mu_i\}, \{\gamma_i^2\}) P(\{s_{ij}\} \mid q_j, \{\mu_i\}, \{\gamma_i^2\}) \\
&= P(q_j) \prod_i P(s_{ij} \mid q_j, \mu_i, \gamma_i^2) \\
&= \mathcal{N}(m_j, s_j^2) \quad \text{where} \\
s_j^2 &= \left(1/\bar{s}^2 + \sum_i 1/\gamma_i^2\right)^{-1} \\
m_j &= s_j^2 \left(\bar{m}/\bar{s}^2 + \sum_i (s_{ij} - \mu_i)/\gamma_i^2\right)
\end{aligned}$$

In other words, the posterior distribution of each q_j is a Gaussian whose mean is the average of the relevant s_{ij} after shifting each one by the learner's bias μ_i and then scaling it by γ_i^2 . We can achieve a non-informative prior by setting the variance \bar{s}^2 to be very high (e.g., 1000).

M-Step: We derive the auxiliary function Q as the expectation, w.r.t. the PDF \tilde{P} computed during the E-Step, of the joint log-likelihood of the observed ratings $\{s_{ij}\}$ and hidden ratings $\{q_j\}$. In the derivation below, C and D are constants that do not depend on any of the parameters.

$$\begin{aligned}
Q(\{\mu_i\}, \{\gamma_i^2\}) &= \mathbb{E} [\log P(\{s_{ij}\}, \{q_j\} \mid \{\mu_i\}, \{\gamma_i^2\})] \\
&= \mathbb{E} \left[\log \prod_j P(q_j \mid \{\mu_i\}, \{\gamma_i^2\}) + \right. \\
&\quad \left. \log \prod_{ij} P(s_{ij} \mid \{q_j\}, \{\mu_i\}, \{\gamma_i^2\}) \right] \\
&= \mathbb{E} \left[\log \prod_j P(q_j) + \log \prod_{ij} P(s_{ij} \mid q_j, \mu_i, \gamma_i^2) \right] \\
&= \sum_j \mathbb{E} [\log P(q_j)] + \sum_{ij} \mathbb{E} [\log P(s_{ij} \mid q_j, \mu_i, \gamma_i^2)] \\
&= \sum_{ij} \int_{-\infty}^{+\infty} dq_j \tilde{P}(q_j) [\log P(s_{ij} \mid q_j, \mu_i, \gamma_i^2)] + C \\
&= - \sum_{ij} \int_{-\infty}^{+\infty} dq_j \tilde{P}(q_j) \left[\frac{(s_{ij} - q_j - \mu_i)^2}{2\gamma_i^2} + \log \gamma_i \right] + D \\
&= - \sum_i \log \gamma_i - \\
&\quad \frac{1}{2} \sum_{ij} \int_{-\infty}^{+\infty} dq_j \tilde{P}(q_j) [(s_{ij} - q_j - \mu_i)^2 / \gamma_i^2] + D \\
&= - \sum_i \log \gamma_i - \frac{1}{2} \sum_{ij} [(s_{ij} - \mu_i)^2 / \gamma_i^2 - \\
&\quad \frac{2(s_{ij} - \mu_i)}{\gamma_i^2} \int_{-\infty}^{+\infty} dq_j \tilde{P}(q_j) q_j + \frac{1}{\gamma_i^2} \int_{-\infty}^{+\infty} dq_j \tilde{P}(q_j) q_j^2]
\end{aligned}$$

where we omitted the constant D in the last line for brevity. The two integrals are the first and second plain moments of $\tilde{P}(q_j)$. The first is the mean of $\tilde{P}(q_j)$, i.e., m_j . The second can be obtained using the fact that the variance $\mathbb{V}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$ for any random variable x . The second plain moment is thus $m_j^2 + s_j^2$. Hence,

$$\begin{aligned}
Q(\{\mu_i\}, \{\gamma_i^2\}) &= - \sum_i \log \gamma_i - \\
&\quad \frac{1}{2} \sum_{ij} \frac{1}{\gamma_i^2} [(s_{ij} - \mu_i)^2 - 2(s_{ij} - \mu_i)m_j + m_j^2 + s_j^2]
\end{aligned}$$

We now differentiate with respect to each parameter, set to 0, and solve:

$$\begin{aligned}
\frac{\partial Q}{\partial \mu_i} &= -\frac{1}{2} \frac{1}{\gamma_i^2} \sum_j (-2(s_{ij} - \mu_i) + 2m_j) \\
0 &= -\frac{1}{\gamma_i^2} \sum_j (\mu_i - s_{ij} + m_j) \\
\sum_j \mu_i &= \sum_j (s_{ij} - m_j) \\
\mu_i &= \frac{1}{N_i} \sum_j (s_{ij} - m_j) \quad \text{where} \\
N_i &\text{ is the \# of ratings from person } i \\
\frac{\partial Q}{\partial \gamma_i} &= -1/\gamma_i + \\
&\quad \frac{1}{\gamma_i^3} \sum_j [(s_{ij} - \mu_i)^2 - 2(s_{ij} - \mu_i)m_j + m_j^2 + s_j^2] \\
\gamma_i^2 &= \sum_j [(s_{ij} - \mu_i)^2 - 2(s_{ij} - \mu_i)m_j + m_j^2 + s_j^2] \quad (2)
\end{aligned}$$

For our experiments we conducted 50 EM iterations.

3.2 Regularizing the model

In the full-fledged GRAM, all of the parameters (bias and reliability of each rater) are learned in an unsupervised fashion (see Section 3.1). Given enough data, these parameters can lead to more accurate estimates of each q_j . However, given limited data, it can also be useful to regularize the model by removing parameters and/or fixing them to known values. In fact, if there are too few subjective scores s_{ij} per learner, then it is important to remove some parameters because otherwise the model encounters identifiability problems. Hence, we considered several variants of the GRAM: (1) each γ_i^2 is estimated, but each μ_i is fixed to 0; (2) each μ_i is estimated, but $\gamma_i^2 = 1$. Finally, we also explored the hypothesis that the students with the higher pre-test scores might, perhaps due to a higher overall engagement, also be more reliable in giving subjective ratings. Hence, we also tried: (4) μ_i is estimated, but $\gamma_i^2 = 1/\sqrt{\mathbb{E}_j[p_{ij}] + \epsilon}$, where $\mathbb{E}_j[p_{ij}]$ is the average (over all their assignments) pre-test score p_{ij} of student i before receiving resource j , and $\epsilon = 0.1$ ensures that the denominator is positive.

4. MODELS FOR COMPARISON

We compared the GRAM to two other models: (1) unweighted average of subjective scores, and (2) prediction model trained directly on pre- and post-test scores.

4.1 Unweighted average of subjective scores

Instead of using the GRAM, we can estimate the quality q_j of each resource j simply as the unweighted average, over all learners who rated j , of their subjective rating scores s_{ij} .

4.2 Average post-test minus pre-test scores

The primary goal of our paper is to assess to what extent subjective scores can estimate the learning gains as measured in a pre-test/post-test paradigm. Hence, a strong baseline – indeed, a likely upper bound – to which to compare our GRAM approach is using a prediction model that *directly* uses test scores (on training data) to estimate students’ learning gains (on testing data). In particular, for each resource j , we estimate $\mathbb{E}_i[l_{ij}]$ – the average difference between post-tests and pre-tests of all students i in the training set who received resource j . We then use this number to predict the average learning gains of resource j in the test set. Obviously, this requires that the adaptive learning system administer pre- and post-tests to learners in order to assess each resource’s quality, and this can be much more time-consuming than simply asking the learner how much she/he likes it. Note that we also considered a prediction model that additionally uses students’ pre-test scores as a co-variate, which could model possible ceiling effects in the tests. However, our results with that model were slightly worse, and hence we do not report them.

5. EXPERIMENT

To assess how well subjective scores of the resources’ quality predicted their associated learning gains, we conducted a randomized experiment on Mechanical Turk. Each participant was paid \$1 and could complete up to 3 tasks. In each task, the pre- and post-tests were the same, but the learning resource was usually different due to random assignment.

5.1 Overview

During the task, participants learned about logarithms. Logarithms are a topic that many adults have learned, but many have forgotten. The topic is hard enough to induce variability in test scores, but easy enough to be learned (or refreshed) in a short amount of time. The learning resources in our experiment comprised a set of tutorial videos on logarithms, most of which were 2-3 minutes long. These resources were authored by different people around the world and collected in a study by Whitehill & Seltzer [16]. Each tutorial explains the solution to one of the math problems that appeared on the pre-test (see Figure 4).

To select videos for our experiment, we watched over 100 candidate tutorial videos collected by [16]. Each video was watched by at least one of the investigators and labeled as either “High Quality,” “Low Quality,” or “Not Acceptable.” Videos labeled as “Not Acceptable” were excluded. To induce some variability in the quality of videos, we chose one “High Quality” video as well as one “Low Quality” for each of the Basic Logarithm problems in the pre-test (see Figure 4), except for a few problems where only one quality level was available. In total, there were $m = 17$ resources (tutorials) that could be assigned; see Figure 3 for examples.

5.2 Protocol

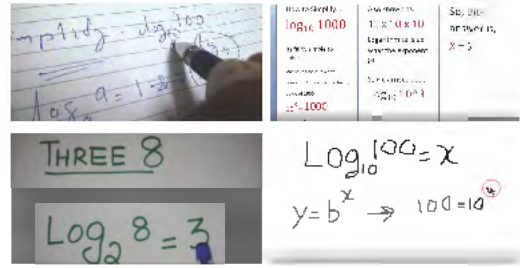


Figure 3: Sample learning resources (tutorial videos on logarithms from [16]) that we used in our experiment.

Basic Logarithms – Simplify:	
$\log_3 1 =$	$\log_9 1 =$
$\log 100 =$	$\log_{\frac{1}{5}} 125 =$
$\log_{10} 1000 =$	$\log_{\frac{1}{x}} x^2 =$
$\log_3 81 =$	$\log_w \frac{1}{w} =$
$\log_2 8 =$	$\log_{\frac{1}{2}} \frac{1}{4} =$
Logarithms and Variables – Simplify:	
$\log_a a^2 =$	$\log_x x^4 =$
$\log_4 4^{2b} =$	$\log_{x-1} (x-1)^y =$
Equations with Logarithms – Solve:	
$\log_3 (x-1) = 4$	$x \log_4 16 = 3$
$z \log_{10} \sqrt{10} = 4$	$y \log_{10} 1000 = 3$

Figure 4: The pre-test on logarithms (borrowed from [16]) in our experiment.

The experiment was built as a web application using HTML and Javascript. Each session consisted of multiple phases:

1. **Survey:** The participants were first asked some basic demographic questions, such as their highest level of education, gender, and age. (Note that we did not use these data in the analyses in this paper.)
2. **Pre-test:** The pre-test surveyed their pre-existing skills in three areas: Basic Logarithms, Logarithms and Variables, and Equations with Logarithms.
3. **Tutorial video:** Participants were then randomly assigned one of the 17 different tutorial videos.
4. **Subjective rating of the resource:** On a Likert scale of 1 to 5, participants were asked how much they agreed with the statement: “This video will help other students learn about logarithms.”
5. **Post-test:** The post-test contained different math problems but was otherwise comparable in format, subject matter, and difficulty to the pre-test.

6. RESULTS AND ANALYSIS

A total of $n = 304$ participants completed the task. Of these, 239 completed 1 task, 35 completed 2 tasks, and 30 completed 3 tasks. Figure 5 shows the box plot, for each resource (tutorial video) j , of the learning gains associated with each resource. There is high variance in learning gains *within* each resource ($\mathbb{E}_j[\mathbb{V}_i[l_{ij}]]$) averaged over the $m = 17$

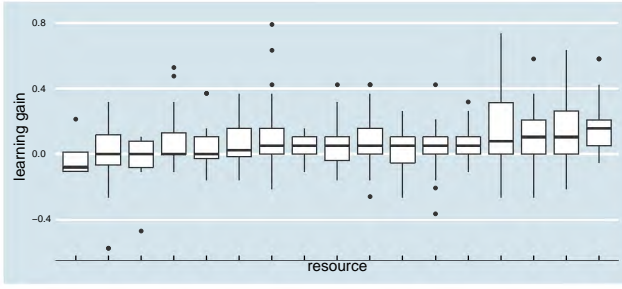


Figure 5: Box plot, for each OER (math tutorial video) j , of the learning gains l_{ij} for all users i who received it. Resources are sorted according to their median learning gains over all learners who received them.

videos is 0.03) that dwarfs the variance in average learning gains *between* resources ($\mathbb{V}_j[\mathbb{E}_i[l_{ij}]]$ is 0.003), where $\mathbb{V}_i[\cdot]$ denotes the variance with respect to learners i and $\mathbb{V}_j[\cdot]$ denotes the variance with respect to resources j . The relative magnitudes of these variances makes the prediction of average learning gains $\mathbb{E}_i[l_{ij}]$ of each individual resource a challenging task.

6.1 Are subjective ratings correlated with average learning gains?

From the set of all subjective scores collected in our experiment, we can aggregate the ratings s_{ij} , using either the proposed GRAM or simply the unweighted average, for each resource j into a quality estimate q_j . Similarly, we can compute the average learning gains associated with each resource j over all students assigned j to obtain $\mathbb{E}_i[l_{ij}]$, where the subscript indicates that the expectation is w.r.t. all students assigned j . This is equivalent to estimating the *average treatment effect* of resource j . We then compute the correlation (Pearson, Spearman) between these two sets of variables. Note that, since many learners in our experiment completed only one task, we needed to simplify the GRAM in order to avoid identifiability problems (see Section 3.2). Hence, instead of the full-fledged GRAM, we used two variants: one where each $\mu_i = 0$, and one where $\mu_i = 0$ and γ_i^2 is determined by the learner’s pre-test score.

Results are shown in Table 1. Because all correlations are estimated within-sample (i.e., there is no separation of training and validation data), computing the p -values (two-tailed) is straightforward. When the GRAM was used to infer only the reliability γ_i^2 (first line of Table 1), the accuracy is low – 0.15 (Pearson) and 0.13. On the other hand, with the other two GRAM variants, when either a bias μ_i for each learner is learned, the performance was much better – up to 0.78 (Pearson) and 0.75 (Spearman) between the inferred q_j and the average learning gains. These results are easily better than what is obtained using just the unweighted average of the learners’ ratings. Estimating γ_i^2 as a function of each learner’s pre-test score did not yield a clear accuracy improvement. Altogether, the results suggest that, with the right aggregation model, learners’ subjective scores carry considerable information about the average learning gains of the resources they receive.

Predicting learning gains within-sample

Method	Pearson	Spearman
GRAM (learn γ_i^2)	0.15 ($p = 0.56$)	0.13 ($p = 0.63$)
GRAM (learn μ_i)	0.78 ($p < 0.001$)	0.70 ($p = 0.002$)
GRAM (learn μ_i , set γ_i^2 from pretest)	0.76 ($p < 0.001$)	0.75 ($p < 0.001$)
Unweighted average	0.38 ($p = 0.14$)	0.54 ($p = 0.03$)

Table 1: Accuracies, and associated p -values, of different models when predicting the average learning gains $\mathbb{E}_i[l_{ij}]$ of the resources from subjective ratings reported by learners. For aggregating learners’ subjective ratings, we consider both the unweighted average as well as the quality scores inferred using the GRAM.

6.2 Do subjective ratings predict the average learning gains for new students?

Suppose some *new* students enter the adaptive learning community. How accurately can we predict the average learning gains $\mathbb{E}_i[l_{ij}]$ of a resource j for these learners? How does this accuracy compare to that of a prediction model in which we estimate the effectiveness of each resource directly based on pre- and post-test data?

We conducted 3-fold cross-validation, where the same students never appear in more than one fold. From the training data in each fold, we use GRAM to infer the latent variables q_j from the subjective scores s_{ij} ; we use the variant in which only μ_i is learned. We then compute the correlation (Pearson, Spearman) between q_j and the average learning gains of resource j over all students i in the test set who received j . Due to the high variability in results over the 3 folds, we repeated the 3-fold cross-validation 30 times, and averaged the results over trials. In each trial, we ensured that the data were randomly partitioned such that every resource was assigned to at least 1 learner in at least 2 folds (i.e., one testing fold and one training fold).

In the cross-validation framework, computing p -values is not straightforward because the estimates from each fold are not statistically independent. Instead, we estimated the uncertainty of each correlation as the average (over the 30 trials) standard error (i.e., the standard deviation of the correlations over the $K = 3$ folds, divided by \sqrt{K}). We compare the accuracy of predictions obtained with the GRAM to the predictions by the *unweighted average* model (Section 4.1), and also to the predictions from a model that has direct access to the *test scores* (see Section 4.2). The latter is a strong comparison because it has access to actual pre- and post-test scores, whereas the other models do not.

Results are shown in Table 2. The GRAM – which utilizes only subjective scores, not test results, of the training data – is able to predict the average learning gains for new learners with higher accuracy (0.49 Pearson and 0.43 Spearman correlation) compared to the model that uses pre- and post-test data (0.36 Pearson and 0.41 Spearman correlation) to estimate the quality of each resource. Even the unweighted average of learners’ subjective ratings retains most of the prediction accuracy that could be achieved using explicit pre-

Predicting learning gains for new students

Method	Pearson	Spearman
GRAM (learn μ_i)	0.49 (± 0.11)	0.43 (± 0.11)
Unweighted average	0.32 (± 0.09)	0.35 (± 0.11)
Predict from test scores	0.36 (± 0.09)	0.41 (± 0.08)

Table 2: Accuracies (\pm their standard errors) over $K = 3$ cross-validation folds, of different models when predicting the average learning gains $\mathbb{E}_i[l_{ij}]$ of *new* learners (i.e., not used for training).

and post-test score data. All in all, our results suggest that (1) learners’ subjective ratings carry considerable information that could be useful in an adaptive learning community for deciding which resources are more effective than others, and (2) using a crowdsourcing consensus model such as our proposed GRAM can potentially yield higher accuracy than simply taking the unweighted average.

7. CONCLUSION

We investigated whether learners’ subjective opinions about the quality of learning resources (e.g., a tutorial video) are correlated with the learning gains (post-test minus pre-test) associated with receiving those resources. This could have implications for adaptive online learning communities in which open educational resources (OER) are served to students based on estimates of how effective they would be for learning: Rather than giving relatively time-consuming pre- and post-tests, the adaptive learning platform could instead simply ask learners how helpful they found the resources to be. We developed a novel Gaussian Rating Aggregation Model (GRAM) with which to aggregate many learners’ subjective scores into an overall quality estimate for each resource. Based on an experiment that we conducted on Mechanical Turk, we found that (1) subjective scores are highly correlated with average learning gains (Pearson correlation of 0.78). Moreover, (2) when predicting the average learning gains for learners who are *new* to the learning community, the accuracy (Pearson correlation of 0.49) using the GRAM from subjective scores was even better than estimating learning gains from test scores.

Future work will consider how to combine subjective scores with test data in order to arrive at improved estimation accuracy of resources’ effectiveness. Moreover, with the goal to personalize education, it would be interesting to explore how to harness subjective ratings to estimate *individual* learning gains rather than just *average* learning gains. Finally, it is important to establish whether the results we collected in our study on adult participants from Mechanical Turk generalizes to more authentic online learning communities (e.g., Khan Academy, ASSISTments).

Acknowledgement and disclaimer: We gratefully acknowledge a grant (N00014-18-1-2768) from the Office of Naval Research that supported this research. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Office of Naval Research.

8. REFERENCES

- [1] Y. Baba and H. Kashima. Statistical quality estimation for general crowdsourcing tasks. In *Knowledge discovery and data mining*. ACM, 2013.
- [2] J. Bragg, D. S. Weld, et al. Crowdsourcing multi-label classification for taxonomy creation. In *AAAI conf. on human computation and crowdsourcing*, 2013.
- [3] R. Bromme, R. Rambow, and M. Nückles. Expertise and estimating what other people know: The influence of professional experience and type of knowledge. *Journal of experimental psychology: Applied*, 2001.
- [4] S. E. Carrell and J. E. West. Does professor quality matter? evidence from random assignment of students to professors. *Journal of Political Economy*, 2010.
- [5] A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, 1979.
- [6] N. T. Heffernan, K. S. Ostrow, K. Kelly, D. Selent, E. G. Van Inwegen, X. Xiong, and J. J. Williams. The future of adaptive learning: Does the crowd hold the key? *International Journal of Artificial Intelligence in Education*, 26(2), 2016.
- [7] P. A. Kirschner and J. J. van Merriënboer. Do learners really know best? urban legends in education. *Educational psychologist*, 48(3), 2013.
- [8] C. Piech, J. Huang, Z. Chen, C. Do, A. Ng, and D. Koller. Tuned models of peer assessment in MOOCs. *arXiv preprint arXiv:1307.2579*, 2013.
- [9] A. N. Rafferty, H. Ying, and J. J. Williams. Bandit assignment for educational experiments: Benefits to students versus statistical power. In *Artificial Intelligence in Education*, 2018.
- [10] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. *Journal of Machine Learning Research*, 2010.
- [11] P. Smyth, U. M. Fayyad, M. C. Burl, P. Perona, and P. Baldi. Inferring ground truth from subjective labelling of venus images. In *Advances in neural information processing systems*, 1995.
- [12] J. G. Tullis. Predicting others’ knowledge: Knowledge estimation as cue utilization. *Memory & cognition*, 46(8):1360–1375, 2018.
- [13] R. K. Vinayak and B. Hassibi. Crowdsourced clustering: Querying edges vs triangles. In *Advances in Neural Information Processing Systems*, 2016.
- [14] D. S. Weld, E. Adar, L. Chilton, R. Hoffmann, E. Horvitz, M. Koch, J. Landay, C. H. Lin, and M. Mausam. Personalized online education—a crowdsourcing challenge. In *Workshops at AAAI*, 2012.
- [15] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, 2009.
- [16] J. Whitehill and M. Seltzer. A crowdsourcing approach to collecting tutorial videos—toward personalized learning-at-scale. In *Learning@ Scale*. ACM, 2017.
- [17] J. J. Williams, J. Kim, A. Rafferty, S. Maldonado, K. Z. Gajos, W. S. Lasecki, and N. Heffernan. Axis: Generating explanations at scale with learnersourcing and machine learning. In *Learning@ Scale*. ACM, 2016.

Early Detection of Wheel Spinning: Comparison across Tutors, Models, Features, and Operationalizations

Chuankai Zhang, Yanzun Huang, Jingyu Wang, Dongyang Lu, Weiqi Fang, John Stamper, Stephen Fancsali, Kenneth Holstein, Vincent Aleven
Carnegie Mellon University, Carnegie Learning, Inc.
{chuankaz, yanzunh, jingyuw1, dongyanl, weiqif}@andrew.cmu.edu,
{jstamper, kjholste, aleven}@cs.cmu.edu, sfancsali@carnegielearning.com

ABSTRACT

“Wheel spinning” is the phenomenon in which a student fails to master a Knowledge Component (KC), despite significant practice. Ideally, an intelligent tutoring system would detect this phenomenon early, so that the system or a teacher could try alternative instructional strategies. Prior work has put forward several criteria for wheel spinning and has demonstrated that wheel spinning can be detected reasonably early. Yet the literature lacks systematic comparisons among the multiple wheel spinning criteria, features, and models that have been proposed, across multiple evaluation criteria (e.g., earliness, precision, and generalizability) and datasets. In our experiments, we constructed six wheel spinning detectors and compared their performance under two different wheel spinning criteria with three datasets. The results show that two prominent criteria for wheel spinning diverge substantially, and that a Random Forest model has the most consistent performance in early detection of wheel spinning across datasets and wheel spinning criteria. In addition, we found that a simple model overlooked by previous research (Logistic Regression trained on a single feature) is able to detect wheel spinning at an early stage with decent performance. This work brings us closer to unifying strands of prior work on wheel spinning (e.g., understanding how different criteria compare) and to early detection of wheel spinning in educational practice.

1. INTRODUCTION

Intelligent tutoring systems (ITS) aim to guide students towards mastery of knowledge components by providing step-by-step personalized guidance. However, there are cases where students persistently work on problems without making progress towards mastery. This phenomenon of unproductive student persistence has been called “wheel spinning” [1]. If ITSs were able to detect potential wheel spinning as early as possible, they might be able to adjust their instructional strategies accordingly to avoid wasting students’ time.

Chuankai Zhang, Yanzun Huang, Jingyu Wang, Dongyang Lu, Weiqi Fang, John Stamper, Stephen Fancsali, Kenneth Holstein and Vincent Aleven “Early detection of wheel spinning: Comparison across tutors, models, features, and operationalizations” In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 468 - 473

Beck & Gong [1] operationalized wheel spinning as failing to get three attempts correct in a row within the first 10 practice opportunities. We refer to this as “three correct in a row criterion.” They presented evidence in [3] that wheel spinning is not a rare phenomenon. Other operationalizations of unproductive persistence have since been proposed. For example, Predictive Stability (PS) is a when-to-stop policy for ITSs proposed in [5], which stops when the probability of a student getting the next step correct stabilizes. The policy uses student performance at each step to decide whether the ITS should stop giving more questions, either because of mastery or wheel spinning. The Predictive Stability++ (PS++) policy [5] provides further analysis about mastery *after* the PS policy would have stopped. This policy can detect wheel spinning under various student models. Although the two operationalizations are, at the surface, rather different, prior work has not investigated how they compare.

More generally, although prior work has introduced various machine learning models for the early detection of wheel spinning, fitted on a variety of datasets, we are not aware of any systematic comparison across models, datasets, and operationalizations. This makes it difficult for researchers to compare and establish global evaluations, which is important both for practical and theoretical reasons. Therefore, in this study, we conducted a comprehensive examination to address the following questions: (1) To what extent do different operationalizations of wheel spinning agree or disagree? (2) Which set of features leads to better predictions? (3) What is the simplest set of features that can be effective? (4) What are some good methods for early detection of wheel spinning? and (5) How early can these methods detect wheel spinning with decent performance?

2. DATASET

We used three datasets in our experiments. Two of the datasets were collected from two sections of Algebra content in the MATHia ITS [9] during the 2017-18 school year. Sections within MATHia, which is built on Cognitive Tutor technology from Carnegie Learning, provide instruction and practice on a series of KCs via multi-step problem solving tasks, each problem providing practice on several KCs (we will refer to this as the “CL1 dataset” and “CL2 dataset”). There are 132,551 student-KC pairs in CL1 dataset and 419,832 student-KC pairs in CL2 dataset. The third dataset is from a high school geometry tutor [6] (we will refer to this as the “Geometry dataset”) with 8175 student-KC pairs.

The datasets used in these experiments were exported from DataShop data [10].

2.1 Label Generation

We labeled each (student, KC) pair in our three datasets according to both the three correct in a row criterion and the PS++ policy [5]. With each criterion, there are three possible label values: *mastered*, *wheel spinning*, and *indeterminate*. A (student, KC) pair was labeled *indeterminate* if there were not enough steps to determine whether the student has or will master the KC. Following [1], we discarded (student, KC) pairs labeled as ‘indeterminate’ before training our models, given insufficient data to apply the criterion.

2.1.1 Three Correct in a Row

We generated labels for the three correct in a row criterion as follows: For each (student, KC) pair, if there were three or more contiguous correct attempts within the first 10 steps, then the (student, KC) pair was labeled *mastered* (even if there were less than 10 steps). The (student, KC) pairs that did not reach mastery with 10 or more steps were labeled *wheel spinning*. The (student, KC) pairs with less than 10 steps and no occurrence of three contiguous correct steps were labeled *indeterminate*. Under this criterion, the frequency of wheel spinning in CL1, CL2 and Geometry dataset is 6.6%, 0.56%, and 10.2% of (student, KC) pairs, respectively.

2.1.2 Predictive Stability++

The second set of labels are derived from the PS++ policy, which is defined as not reaching a mastery condition *after* a student model’s predictions of next step correctness have stabilized to a steady state [5]. In our analysis, we used Bayesian Knowledge Tracing (BKT) as the student model. On the Geometry dataset, we used BKT parameters obtained by fitting the model to data. In the other two datasets, we used the “shipped parameters,” that is, the parameters actually used by the ITS. For each step in a (student, KC) pair, BKT calculates $P_C(t)$, which is the probability of getting a correct response for the current step, as well as $P_{C|0}(t) = P(C_{t+1}|¬C_t)$ and $P_{C|1}(t) = P(C_{t+1}|C_t)$, the probabilities of a correct response on the next step, conditioned on a correct or incorrect response on the current step. When $P_{C|0}(t)$ and $P_{C|1}(t)$ converge, the stopping criterion defined in PS [5] is reached. We then determine the label of the (student, KC) pair as follows: According to PS++ [5], after convergence, when $P_C(t)$ is close enough to its upper bound, we consider the student has mastered this KC. Otherwise the (student, KC) pair is labeled as wheel spinning. For those (student, KC) pairs where the stopping criterion has never been met, we assign *indeterminate* as the label. Under this criterion, the frequency of wheel spinning in CL1, CL2 and Geometry dataset is 24.2%, 2.17%, and 13.2% of (student, KC) pairs, respectively.

2.2 Features

As we explored ways of creating early detectors for wheel spinning, we used a total of 28 features. Among these, 15 were introduced by [3]. These 15 features are extracted to analyze and record three aspects of students’ learning progress. The first aspect is students’ learning performance like ‘correct response count’ and ‘prior problem count with

hint request’, which indicate whether the student is doing well on a particular KC. The second aspect is the ‘seriousness’ of students, including ‘prior problem fast correct’ and ‘prior problem slow incorrect’. These features indicate whether the student appears to be making a deliberate effort on a particular KC. The third category of the features includes general features like ‘skill id’. In addition, we used 7 features introduced by [4], and 6 new features based on our previous research and our explorations on the Carnegie Learning dataset. A complete list of features and their descriptions can be found in the online appendix.¹

3. EXPERIMENTS AND DISCUSSION

We conducted the following experiments and analyses to answer the research questions listed in section 1.

3.1 Compatibility of Operationalizations

3.1.1 Comparing Operationalizations

Regarding Research Question (1) (to what extent do different operationalizations of wheel spinning agree or disagree?), a first observation is that the overall frequency of wheel spinning, reported above, differs substantially under the two operationalizations, with no clear pattern of one predicting more wheel spinning than the other. The confusion matrices (Figure 1, 2, 3) that compare the two operationalizations on each of the three datasets provide further insight into this divergence. For instance, in the CL1 dataset, among all (student, kc) pairs that are labeled as wheel spinning by either operationalization, the two operationalizations agree on only 22.2% of them. In CL2 dataset, the same wheel spinning agreement percentage is 14.1%. In the Geometry dataset, it is 41.6%. The agreement on wheel spinning is generally less than 50%.

Three Correct in a Row	Predictive Stability		
	Master	Indeterminate	Wheel Spinning
Master	31.15%	13.88%	10.01%
Indeterminate	0.23%	29.51%	8.60%
Wheel Spinning	0.32%	0.69%	5.61%

Figure 1: Comparison of two different criteria for wheel spinning in the CL1 dataset.

Three Correct in a Row	Predictive Stability		
	Master	Indeterminate	Wheel Spinning
Master	84.84%	2.35%	1.43%
Indeterminate	0.95%	9.48%	0.40%
Wheel Spinning	0.12%	0.10%	0.34%

Figure 2: Comparison of two different criteria for wheel spinning in the CL2 dataset.

To investigate the divergence between the two operationalizations in more detail, we present examples of (student, KC) pairs where the criteria disagree (see Figures 4 and

¹<https://tinyurl.com/edm19supplement>

Three Correct in a Row	Predictive Stability		
	Master	Indeterminate	Wheel Spinning
Master	55.96%	7.14%	1.24%
Indeterminate	10.26%	10.12%	5.05%
Wheel Spinning	1.31%	2.06%	6.86%

Figure 3: Comparison of two different criteria for wheel spinning in the Geometry dataset.

5). These visualizations show the student first attempted response on each step together with different wheel spinning metrics. Specifically, the student’s correct answers, incorrect answers, and hints for the given KC are visualized at the bottom of each graph. When there are 3 contiguous green dots within the first 10 steps (shown in a dashed-lined box), the KC will be considered mastered under the three correct in a row criterion. Shown above are $P_C(t)$, $P_{C|0}(t)$, $P_{C|1}(t)$, P_{master} , and upper bound of $P_C(t)$. A vertical, dashed line shows the stopping step for PS; the x-axis denotes the practice opportunity using a 0-based index. In most of the cases in which the three correct in a row criterion and PS++ agree with each other, there is a clear pattern in students’ responses towards mastery or wheel spinning. For example, many contiguous corrects will generally result in mastery under both criteria; many contiguous incorrect attempts will result in agreement for wheel spinning. Figure 4 shows one case where PS++ detects wheel spinning but the three correct in a row criterion detects mastery. For this specific (student, KC) pair, mastery (under both criteria) occurs past the point where PS stopped; the initial string of incorrect responses appears to have been influential.

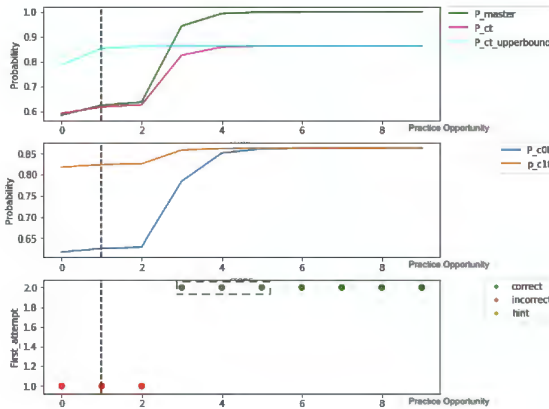


Figure 4: An example of (student, KC) where the three correct in a row criterion detects mastery and PS++ detects wheel spinning.

Figure 5 shows the opposite situation, where PS++ gives a mastery label but the three correct in a row criterion is detecting wheel spinning. In this instance, mastery under the various criteria happens past the 10 step cutoff.

3.1.2 Discussion

We found some overlap but also substantial disagreement between the two operationalizations of wheel spinning, the

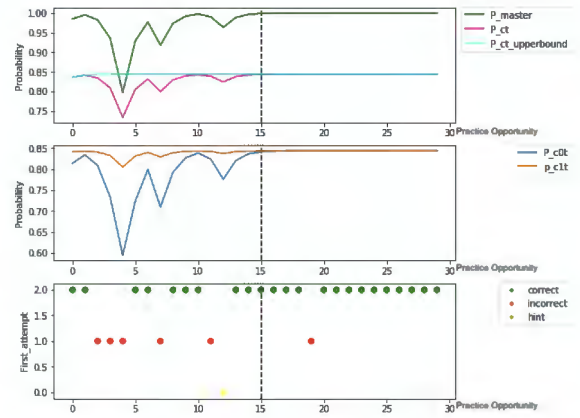


Figure 5: An example of (student, KC) where the three correct in a row criterion detects wheel spinning and PS++ detects mastery.

three correct in a row criterion and PS++. The operationalizations tend to agree when the student’s performance is obvious and steady (e.g. the student is doing extremely well or poorly on a KC, or when, as is common, there is a gradual increase in performance). However, these criteria can disagree when students’ responses fluctuate. One of the reasons is that the two operationalizations judge mastery in different ways. The three correct in a row criterion, with the explicit 10-step (or other configurable number of steps) cutoff, focuses on mastery in the early stage. Student performance after the cutoff is not taken into account. In contrast, PS++ may consider long-term performance; its mastery judgment can be made using more data, although, as seen in one of our examples, PS++ may stop too early on KCs with lower P_{learn} and an early string of incorrect responses.

3.2 Feature Effectiveness

In this section, we aim to explore the effectiveness of features we are using. In particular, we focus on Research Questions (2): Which set of features lead to better predictions; and (3): What is the simplest set of features that can be effective?

3.2.1 Feature Importance with Random Forest

In order to find a set of features for better prediction, we trained a Random Forest model and generated the feature importance graph. Feature importance of a Random Forest is measured by the total decrease in Gini impurity averaged over all the trees in the ensemble [2]. We rely on [8]’s implementation of Random Forest and feature importance and used the default hyperparameters. Figure 6 is an example of the feature importance graph. Among the set of 28 features, four are repeatedly identified as important by the Random Forest model. In all 6 scenarios (three datasets with two wheel spinning criteria), at least three of the four selected features appear in the top five most important features. Two of these features are related to students’ performance: ‘Correct Response Count’ and ‘Correct Response Percentage’. The other two features are related to the speed and attentiveness of students: ‘Exp Mean Response Time Z-Score’ and ‘Prior Step Count Normal Correct’.

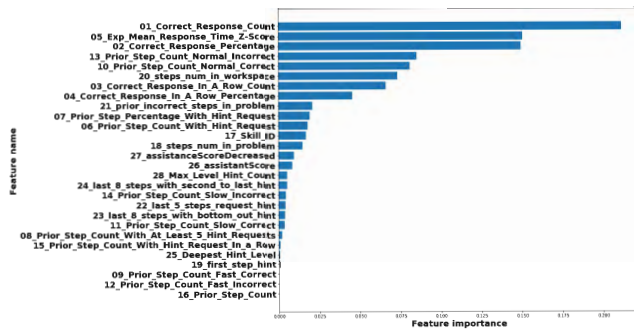


Figure 6: An example of feature importance, trained on CL1 dataset under PS++ criterion.

3.2.2 Fitting Single-feature Logistic Regression

The predictive power of very simple wheel spinning detectors is under-explored in prior literature. Here, we are interested in finding out the predictive value of individual features. Therefore, we picked 6 features that were repeatedly deemed as important in the work reported in section 3.2.1 and built 6 Logistic Regression models, each trained on one of the 6 features. We used [8]’s implementation of Logistic Regression and applied the default hyperparameters.

The results show that the detectors trained on ‘Correct Response Count’, ‘Correct Response Percentage’, ‘Correct Response in a Row Count’ and ‘Prior Step Count Normal Correct’ achieve high precision and recall when predicting the PS++ label. For example, a Logistic Regression model, trained with ‘Correct Response Percentage’ as the single feature, detected wheel spinning with 93.5% precision and 77.1% recall after just 4 steps. Generally, features involving correctness seem to be highly effective in wheel spinning detection. In addition, although the detector trained on ‘Assistance Score’ (i.e. the sum of the number of errors and the number of hints on a step) somewhat surprisingly didn’t perform as well as the rest, it still reached 68.4% precision and 60% recall in the fourth step.

3.2.3 Discussion

In our experiment, we found that the features that involve correctness of steps tend to be effective in predicting wheel spinning, independent of the criteria or datasets used. These results make sense because intuitively, if a student can get a large percentage of steps correct, then this student is likely on their way to mastering the given KC. They also lead to the question of whether we can build an effective detector that relies on correctness only. The results above show that a Logistic Regression model trained with ‘Correct Response Percentage’ is able to give us comparable result to other models, although it suffers more from the cold start problem and fluctuates more than other detectors. In addition, other aspects of the step-solving process, including time and help requested, can also be useful, as ‘Exp Mean Response Time Z-Score’ and ‘Assistance Score’ also have high feature importance while training Random Forest model. These findings indicate that certain aspects in students’ learning performance help predict wheel spinning regardless of the problem setting and tutoring system.

3.3 Early Detection Models

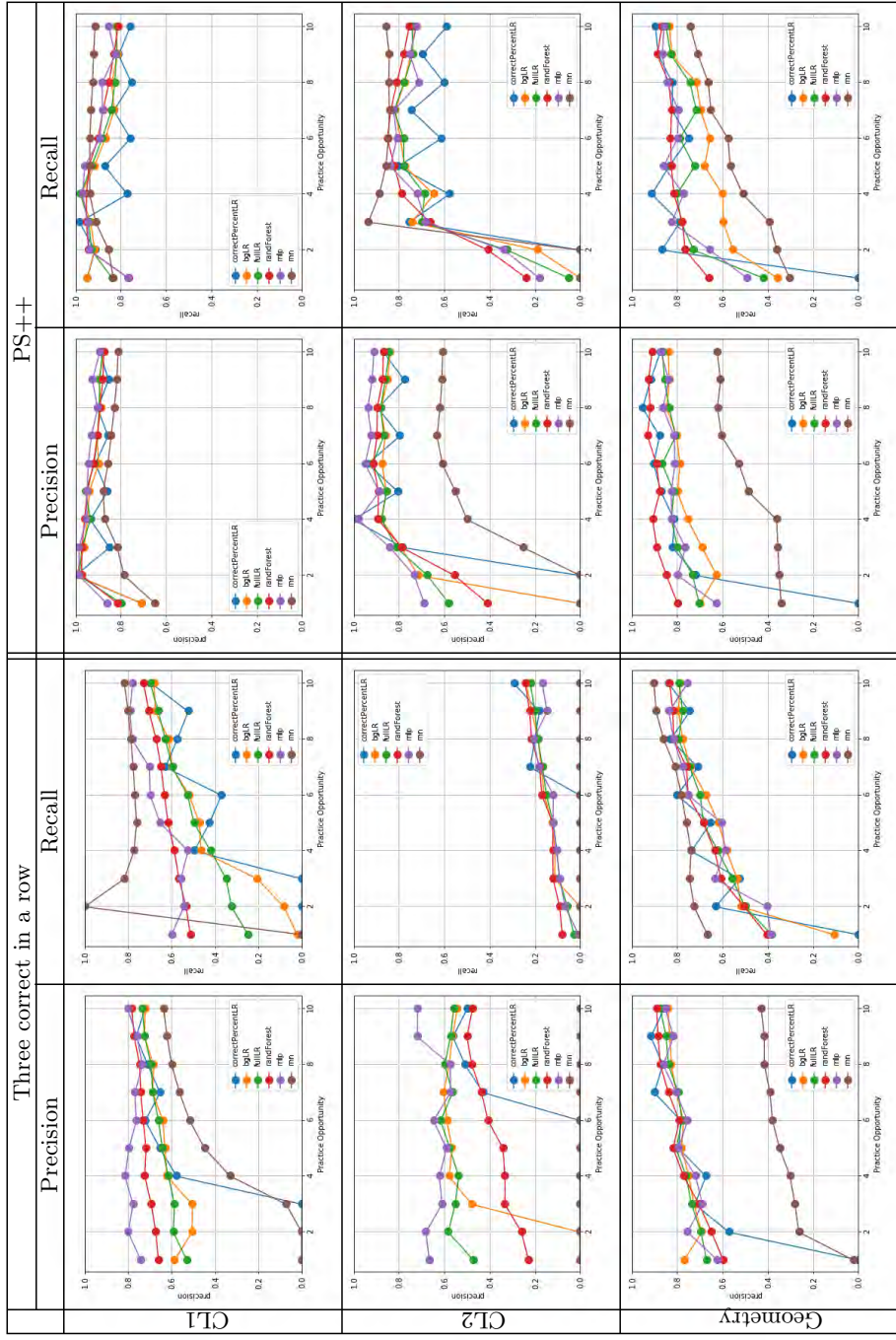
To answer Research Question (4) and (5), we trained multiple machine learning models to study their performance on early detection. We used a Logistic Regression model, trained on the same set of features as in [1], as a baseline. Another detector based on Logistic Regression was trained with the full set of 28 features. In addition, we include one of the detectors used in section 3.2.2, namely, a Logistic Regression model trained on ‘Correct Response Percentage’, to compare the performance of a simple model with that of other more complex ones. Inspired by [4], we also included a Random Forest model. Finally, we trained two neural-network-based detectors: a 5-layer fully-connected artificial neural network and a 3-layer LSTM. We split our dataset into training and testing data with a 6:4 ratio.

In order to study how early the detectors could accurately detect wheel spinning - by early we mean early in the opportunity count for any given (student, KC) pair - we fitted Random Forest, Logistic Regression and MLP models separately for each practice opportunity - that is, separately with data up to and including opportunity N , for N from 1 to the available data for the given (student, KC) pair. (The labels were computed based on all data, as described above.) Doing so was necessary for these three models as they do not have a recurrent structure to handle variable length steps and most of our features are accumulative. By contrast LSTMs are inherently recurrent, so we trained it on data from every step. We rely on [8]’s implementation of Logistic Regression, Random Forest and Multi-Layer Perceptron (MLP) and [7]’s implementation of LSTM. For Logistic Regression and Random Forest, we used the default hyperparameter provided by [8]. For MLP, we had 3 hidden layers with 64, 32, 16 units, respectively. For LSTM, we used 2 layers with 64 hidden units. For each dataset, we evaluated our detectors on the corresponding test data, and computed precision and recall for the wheel spinning class. Figure 7 shows the detectors’ performance on early detection, with two (precision, recall) plot pairs for each detector on the ‘wheel spinning’ class, one pair for each of the two operationalizations.

3.3.1 Model Performance

We found that these detectors in general perform well under the PS++ criterion. Most of them reached more than 60% precision and recall after the fourth step, and more than 80% precision and accuracy after the sixth step. Under the three correct in a row criterion, the detectors in general perform worse in terms of both precision and recall compared to the PS++ criterion. In particular, in CL2 dataset, we observed extremely low recall for all models. As foreshadowed in section 3.2.2, the single-feature Logistic Regression model (blue line in Figure 7) achieved decent performance compared to more complex models, except on the CL2 dataset. Although it tends to have lower precision and recall in earlier steps, after step 4, its performance is comparable to those of Logistic Regression trained on 15 features and Logistic Regression trained on the full feature set.

The two most accurate models are Random Forest (red line in Figure 7) and MLP (purple line in Figure 7). In particular, at the fourth step, Random Forest exhibits (77.2%, 63.5%) precision and recall on three correct in a row and



the baseline model achieves (74.9%, 58.2%) and (75.3%, 59.9%). To support comparison, we kept the hyperparameters consistent across all situations. However, if one is interested in getting even better performance, we recommend further tuning these parameters based on specific situation.

3.3.2 Discussion

The models differ in how well they generalize across datasets and wheel spinning criteria. Among them, Random Forest and MLP were those with the most competitive results. Besides its performance, Random Forest also provides an evaluation of feature importance (as in section 3.2.1) and possibly greater interpretability. In addition, to our surprise, we found that the detector trained on a single feature related to correctness consistently produced results not far behind from other detectors. Features unrelated to correctness, on the other hand, are more dependent on the context, such as differences across ITSs and usage, or difficulty of the underlying domain. However, the downside of using a single correctness-related feature is that correctness only measures one (albeit important) aspect of the students’ behavior while solving steps. Detectors like this may overlook the fact that some students can learn from solving steps and reach mastery slightly later (e.g. getting the first four steps

wrong but answering correctly on the next four) in the process. Nonetheless, this shortcoming should not undermine the power of this model. We recommend its use as a baseline model in future research.

We also find that the detectors perform worse in general under three correct in a row criterion. Even in CL2, where over 97% (student, KC) pairs are under mastery class after discarding the indeterminate class, their performance are extremely low under three correct in a row criterion while decent under PS++ criterion. We failed to come up with a definite explanation to such a phenomenon, but we hypothesize that three correct in a row criterion may not need complicated features to predict, so adding new features merely introduces noise for detectors. Another common issue we found is that these models suffer from the “cold start” problem in every dataset, model, and operationalization. Almost all models had lower than 20% precision and recall in the first three steps in all datasets and metrics on the three correct in a row criterion. This is understandable, since at the first two or three steps, the features collected are often insufficient to determine whether a student has reached mastery or is wheel spinning.

4. CONCLUSION AND FUTURE WORK

In the current work, we aim to move toward clarity and unity in investigations of early detection of wheel spinning. Prior

investigations have not compared across different models for early detection across datasets and operationalizations. To begin addressing this gap, we compared two prominent operationalizations of wheel spinning ([1] and [5]) and compared, across three datasets, the performance of several detectors that were trained with different sets of features. First, the frequency of wheel spinning across the three datasets, 0.56%-10.2% under 3-in-a-row and 2.17%-24.2% under PS++ was in line with what previously studies have reported. For example, in the two datasets used by [3], the wheel spinning ratio was 16% and 6%. Further, we found that two well-known operationalizations of wheel spinning diverge substantially in our three datasets, which is not desirable from either a practical or a theoretical perspective, as we do not know which one to apply or build on. Some typical cases on which they do not agree include getting more KC opportunities correct after [1]’s threshold and answering several consecutive steps incorrectly early followed by much improved performance on later steps. We also found that our models predict PS++ more accurately than they predict [1]’s criterion, most dramatically in the CL2 dataset, but also in the other two data sets (see section 3.3). This finding is surprising especially if one considers that our feature sets included features engineered for prior detectors of three correct in a row reported in the literature [1, 3, 4]. Therefore, it appears unlikely that the more accurate prediction of PS++ is just an artifact of the particular choice of features. What may play a role is that the three correct in a row criterion, in contrast to PS++, does not allow for the fact that KCs vary in difficulty (i.e., require different numbers of practice opportunities, on average, to reach mastery), as is by now well-established [3]. It would be highly desirable to investigate whether our key findings, the discrepancy of the two criteria and the more accurate prediction of PS++, are replicated in a wider range of datasets.

Further, when we evaluated the predictive value of 28 features, we found there is a set of common features that are effective across different datasets and criteria, namely those having to do with step correctness and assistance gained from the system. These features are essential aspects of the learning process to capture for early detection of wheel spinning. Of the 6 wheel spinning detectors we built, the Random Forest model performed consistently well across datasets and operationalizations. Combined with its ability to evaluate feature importance and its potential for interpretability, we recommend trying Random Forest when developing a wheel spinning detector for a tutoring system. In addition, to our surprise, a Logistic Regression model trained on a single feature ‘Correct Response Percentage’ achieved results that were not far behind those of more complex models like MLP. Addressing our research question of how early we could predict wheel spinning, our best model was able to make predictions with decent precision and recall as early as step 4, namely, on CL1 dataset. These results compare favorably with the accuracy achieved by prior early detectors for wheel spinning, discussed in the introduction. Note that we are not trying to recommend using the fourth step as a definitive criterion for earliness. We merely note that, as our results show, machine-learned models can do well from that particular step onward.

There are several limitations of our work that could be studied further in future work. For example, it would be worthwhile to continue to study agreement and disagreement of additional operationalizations of wheel spinning (e.g., [6]), given that the two we studied agreed to a lesser extent than expected. In addition, some hyperparameters in PS++ can be tuned for a given system, which may further support comparisons.

5. ACKNOWLEDGEMENTS

This work was supported by IES Grants R305A180301 and R305B150008. The opinions expressed are those of the authors and do not represent the views of IES or the U.S. ED.

6. REFERENCES

- [1] J. E. Beck and Y. Gong. Wheel-spinning: Students who fail to master a skill. In *AIED*, 2013.
- [2] L. Breiman, J. Friedman, C. Stone, and R. Olshen. *Classification and Regression Trees*. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984.
- [3] Y. Gong and J. E. Beck. Towards detecting wheel-spinning: Future failure in mastery learning. In *Learning @ Scale*, pages 67–74, New York, NY, USA, 2015. ACM.
- [4] S. Kai, M. V. Almeda, R. Baker, C. Heffernan, and N. Heffernan. Decision tree modeling of wheel-spinning and productive persistence in skill builders. *JEDM*, pages 36–71, 2018.
- [5] T. Käser, S. Klingler, and M. Gross. When to stop?: Towards universal instructional policies. In *LAK*, pages 289–298, New York, NY, USA, 2016. ACM.
- [6] R. Liu and K. R. Koedinger. Towards reliable and valid measurement of individualized student parameters. In *EDM*, pages 135–142, 2017.
- [7] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
- [8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *JMLR*, 12:2825–2830, 2011.
- [9] S. Ritter and S. Fancsali. Mathia x: The next generation cognitive tutor. In *EDM*, 2016.
- [10] J. Stamper, K. Koedinger, R. S. d Baker, A. Skogsholm, B. Leber, J. Rankin, and S. Demi. Psic datashop: A data analysis service for the learning science community. In *ITS*, pages 455–455. Springer, 2010.

Detecting Suggestions in Peer Assessments

Gabriel Zingle, Balaji Radhakrishnan, Yunkai Xiao, Edward Gehringer, Zhongcan Xiao, Ferry Pramudianto, Gauraang Khurana, and Ayush Arnav

Department of Computer Science
North Carolina State University
Raleigh, NC 27606
+1 919-515-2066

{gzingle, bradhak, yxiao28, efg, zxiao2, fferry, gkhuran, aarnav}@ncsu.edu

ABSTRACT

Peer assessment has proven to be a useful strategy for increasing the timeliness and quantity of formative feedback, as well as for promoting metacognitive thinking among students. Previous research has determined that reviews that contain suggestions can motivate students to revise and improve their work. This paper describes a method for automatically detecting suggestions in review text. The quantity of suggestions can be treated as a metric for the helpfulness of review text. Even before a review is submitted, the system can tell a reviewer when a review is lacking in suggestions and consequently advise that they be added. This paper presents several neural-network approaches for detecting suggestions and compares them against traditional natural language processing (NLP) methods such as rule-based techniques, as well as past machine-learning approaches.

Our network-based classifiers outperformed rule-based classifiers in every experiment. Our neural-network classifiers attained F1-scores in the low 90% range, outperforming the support vector machine (SVM) classifier whose F1-score was 88%. The naïve Bayes (NB) classifier had an F1-score of 84% and the rule-based classifier had an F1-score of 80%. As in other domains such as determining sentiment, we found that neural-network models perform better than the likes of naïve Bayes and support vector machines when classifying suggestions in text.

Keywords

Peer assessment, suggestion mining, classification techniques, text analytics, text mining.

1. INTRODUCTION

Peer assessment is known to have several advantages for student learning [1]. It provides students with prompt and rich feedback that helps them improve their performance and learning readiness [2]. It gives students an opportunity to learn from others by observing their approaches to solving a problem. Not only do students learn from the feedback they receive; they also benefit from reviewing others. In fact, they probably learn more from reviewing than they do from receiving feedback [3, 4, 5, 6, 7, 8]. Finally, reviews from other students may help instructors to assign more informed grades.

Gabriel Zingle, Balaji Radhakrishnan, Yunkai Xiao, Edward Gehringer, Zhongcan Xiao, Ferry Pramudianto, Gauraang Khurana and Ayush Arnav "Detecting suggestions in peer assessments" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 474 - 479

However, these advantages can only be achieved with high-quality feedback. Nelson and Schunn [2] showed that high-quality feedback comprises several features, including suggestions on how to address problems in the work. Having concrete suggestions makes the feedback actionable for the reviewee and also trains the reviewer to solve problems [2], instead of just focusing on the existence of the problems.

So it is desirable for peer reviews to contain suggestions, but it is not easy for the instructor to give students credit for making them. The instructor would have to look through each review and keep a tally. If this could be done automatically, particularly before the reviewer submits a review [9, 10], it could help reviewers to improve the quality of their feedback, which in turn would help reviewees to improve their work.

2. LITERATURE REVIEW

Before digging in further, we must first decide what constitutes a suggestion. Negi and Buitelaar [11] state that, due to the variation in the definition of suggestion, previous results may be incompatible with each other. In this paper, we define suggestions as comments that constitute advice for making improvements [12].

Suggestions normally contain some or all of the components of specificity [2]: locating the problem, identifying the problem, offering a solution to the problem. Specificity has proven to have a direct correlation to understanding. Understanding is found to be the only significant mediator that directly contributes to the likelihood of implementing the feedback, thus improving the student's work.

Some effort has been made to detect suggestions through conventional natural language processing. These NLP approaches usually utilize rules that match the feedback to a set of predefined linguistic patterns, as well as part-of-speech (POS) tagging and a carefully crafted thesaurus relevant to each domain on which the approach is going to be implemented [13, 14, 15]. The main drawback of these approaches is that they require the knowledge of engineers to define the rules and patterns in the software logic. However, it is almost impossible to foresee all possible rules and patterns that indicate the existence of suggestions in a text document. Thus, as the number of patterns and rules grow, it becomes very difficult for engineers to maintain the software.

Another approach that has shown promising results for detecting patterns is based on machine-learning techniques. For instance, researchers have utilized machine-learning methods such as SVM with some degree of success to find patterns that indicate the sentiment polarity of a text [9].

Machine-learning algorithms are able to discover patterns and rules automatically based on training samples [16]. For software

engineers, this is a game-changer since it allows them to keep the complexity of the software consistent and manageable. There are many machine-learning algorithms that can be used for classifying text as to whether it contains suggestions or not, such as decision trees (DT) and support vector machines [17]. Machine-learning approaches can usually outperform rule-based approaches once they are trained with sufficient and representative samples that have been labeled. However, obtaining labeled data is usually very expensive.

Recently, the neural network approach has stood out for its ability to solve classification problems in different domains, e.g., image and voice recognition, and text classification [11]. This work tries to apply neural networks to suggestion detection and compares the results with the rule-based and traditional machine-learning classifiers.

3. DATA

In this experiment, we used a balanced dataset of peer reviews with a total of 3878 peer reviews, each labeled as containing a suggestion (1) or as not containing a suggestion (0). It is extracted from the Expertiza [18, 19] platform, which we would describe later on in this section. The dataset has two main components: (i) the input text, and (ii) a label indicating whether suggestions were present in the text. The datasets for the neural network classifiers were broken down into an 80–10–10% split for training, validation, and test sets. For the naïve Bayes classifier and the support vector machine classifier, split for training and test sets was 90-10%. All the models were trained and tested on the same dataset.

The platform we mentioned in the previous paragraph, Expertiza, is a web-based peer-review system that allows students to exchange ideas and build shared knowledge. It facilitates anonymous reviews for students' submissions. In this paper, we use reviews submitted to Expertiza to test our approaches. Authors who received reviews manually labeled 15,067 review comments as to whether they contained a suggestion. The labeling of these reviews was incentivized by giving extra credit to students in a class who tagged review comments. A team made up of the instructor, TAs, and authors of this paper went through the labels and removed labels assigned by students whose labels were clearly not trustworthy. Untrustworthy labels included random labels unrelated to review content, all no's or all yes's on reviews with obvious variety of status of containing suggestions. Inter-rater reliability of the dataset was calculated between each team of students labeling peer reviews with a Krippendorff alpha [20] of 0.69 before removing unreliable labels and 0.74 afterwards. Students receiving these reviews were an ideal choice for annotating these reviews, since the reviews were of their own work. Thus, they were capable of judging whether they really contained suggestions. In the preprocessing stage, entries that are invalid to the experiment, such as blank entries, duplicated entries, entries with unrecognizable symbols and marks, as well as html tags, were striped from the dataset. After preprocessing, the dataset contained 5,842 entries without suggestions, and 1,939 entries with suggestions.

With an imbalanced dataset, the process of machine learning is greatly compromised. Classifiers such as naïve Bayes that have been trained on too great a prevalence of a single label type can create a classifier that will only predict that single label. Machine-learning algorithms tend to focus much more on a prevalent class and much less on rare cases, even if the rare cases are trustworthy [21].

Table 1: Sample review comments.

Review Comment	Contains a suggestion?
"Very well written and very obvious how much of an impact the refactoring had."	No
"Yes, all the functionality are covered in the design document."	No
"The test plan talks about automated tests, but then goes on and talks about 2 manual testing scenarios. Details for automation test cases are missing. If the team does not plan on writing automated test cases, then it should be mentioned in the documentation. Otherwise, the team should provide more details about the test plan for automated test cases."	Yes
"Code is good. Just in tests, the code could've been reduced in some cases where same object is being mocked multiple times. "	Yes

One way to deal with this problem is to apply selection techniques. A well-known method is to down-sample the dataset. One should always keep all rare positive samples that need to be focused on and only prune out negative samples [22].

We applied this technique on our dataset in order to minimize the impact of an imbalanced dataset. Down-sampling kept all 1,939 of the positive cases in our dataset and randomly selected 1,939 negative cases to form a new balanced dataset that we used for analysis. Table 1 shows sample review comments with their respective labels. The dataset is available upon request.

4. METHODOLOGY

This section describes the methodology for each classifier. Our input data for these Classifiers is document level review text instead of sentence level, frequently we would observe students describing their solutions in sentences or even paragraphs, and we believe keeping them in their original form would be beneficial for the study. Currently, we are rating the quality of a review based on the presence of a single suggestion. Later on, we can generalize our approach to count the number of sentences that contain suggestions.

4.1 Traditional Machine Learning and Rule-based Methods

Section 4.1 discusses the formation of the rule-based, naïve Bayes, and support vector machine classifiers. Text preprocessing in the form of stop-word removal and stemming was not used except for stemming the data for the naïve Bayes classifier. The reason behind this was due to decreased classifier performance by experimenting with these preprocessing techniques and then testing the classifiers.

4.1.1 Rule-based NLP Methods

Part-of-speech tagging was used to determine the word class of each processed word. The relevant tags used for this classifier included MD (modal auxiliary), VB (verb, base form), VBZ (verb, present tense, 3rd person singular), VBP (verb, present tense, not 3rd person singular), NNS (noun, common, plural), and NNP (noun, proper, singular).

The idea and rules for this rule-based classifier were derived from Gottipati et al. [17]. We used pattern-matching and part-of-speech tagging methods, based on the work of Brun and Hagège [14], Bird et al. [13], and Marcus et al. [15]. The patterns we used are as follows.

- The first rule utilized a pattern-matching technique that locates specific phrases and keywords that indicate the likely presence of a suggestion. The phrases used included “have more”, “suggestion”, “perhaps”, and “better if”.
- The second rule uses part of speech tagging through the Natural Language Toolkit (NLTK) [13] to find a sequence of verb pairs in a review. If a word was tagged as a modal (MD) and the word directly following this was tagged as either of the verb types VB, VBZ, or VBP, then the review would be classified as a suggestion. For example, in the sentence “You should have included a diagram to visualize your results.” the words “should have” would be tagged as MD and VB respectively. This sentence would be classified as a suggestion.
- Following Gottipati, the third rule also used POS tagging. Reviews were classified as a suggestion if a word tagged NNS or NNP was followed by a word of type VB, VBZ, or VBP. This is similar to rule 2 and it was used due to the fact that some words would be labeled with different tags depending on whether they were the first word in a sentence or were located within the sentence. An example of this rule would be the processing of a sentence such as “Could include more examples in your lab report.” where the words “Could include” are tagged as NNP and VBP respectively by the NLTK POS tagger. It is important to note that a word such as “Could” will be tagged as NNP or MD depending on whether it starts a sentence or not. This would classify the sentence as a suggestion. Rule three was later removed from the classifier due to decreased performance as discussed in the results section.

4.1.2 Naïve Bayes

The Naïve Bayes classifier was formed using a train-and-test split of 90-10%. The functionality for creating this classifier is provided by libraries from Scikit-learn [23]. A pipeline of transformers was used to facilitate the construction of the classifier. These transformers include a count vectorizer that converted a collection of text documents to a matrix of token counts for future use. Then the count matrix was transformed into a normalized tf-idf representation to form the linguistic features. These fractional counts from the tf-idf representation then enabled the naïve Bayes classifier to fit the model. As previously noted, stemming the data did result in a slight increase in classifier performance, therefore composing part of the preprocessing for the final model.

4.1.3 Support Vector Machine

The support vector machine classifier was formed using a similar approach to that of the naïve Bayes classifier. The train-and-test split was 90-10%. The same count vectorizer and tf-idf transformer were used to prepare the data as with the naïve Bayes model. Scikit-learn’s [23] stochastic gradient descent training for a linear model was then used to form the support vector machine classifier.

4.1.4 Tools: Traditional Machine Learning and Rule-based

The Python natural language toolkit [13] was used for tokenization and part-of-speech tagging to prepare the text for analysis by the rule-based classifier. Stop-word removal from NLTK was attempted in the rule-based and naïve Bayes classifier, though later abandoned due to the exclusion of vital words for finding patterns

for suggestions, which decreased classifier performance. The Python library Pandas [24] was used to read the dataset into a data frame for reference within the program. Scikit-learn [23] was also used to calculate and present the resulting F1-score for the classifiers. Scikit-learn provided feature extraction and linguistic-feature processing for the naïve Bayes and support vector machine classifiers. The Scikit-learn functionality utilized included a count-vectorizer and tf-idf transformer to prepare the data for classification by these two classifiers.

4.2 Artificial Neural-Network Methods

Neural networks are a state-of-the-art method when it comes to classification. They generally tend to perform well across domains as well as across different types of data such as images, text, and speech. The core idea behind this work is to convert the task of detecting suggestions into a text-classification task and using neural networks to perform this classification. Neural networks work well for this kind of text classification. This is especially true for any kind of recurrent model, such as RNN or LSTM. Recurrent models are widely used for processing any kind of sequential data such as text, speech, and patterns. This is due to their inherent ability to process data in a sequential manner, one step of the sequence at a time. Thus, they are an ideal choice for tasks involving text, such as text classification and text generation.

In this paper, we have focused on utilizing recurrent models to perform text classification. More specifically, we focus on LSTMs and bi-directional LSTMs to achieve the best performance. In addition to the aforementioned recurrent models, we also explore the feasibility of using convolutional neural networks (CNNs) to perform the same tasks. CNNs have traditionally been used to deal with images, but there has been a recent trend toward using them for text-classification tasks. CNNs are not as optimized as recurrent models are for inputs in the form of a sequence, but they offer other benefits such as significantly improved model training times. CNNs can be used either on their own or in conjunction with recurrent models such as LSTMs. We implemented both variants. As is the norm with applying CNN’s to text classification tasks, we used a 1-D CNN everywhere in this paper.

4.2.1 LSTM

LSTM networks are a type of RNNs (Recurrent Neural Networks). They improve upon RNNs by avoiding their single biggest pitfall, the inability to capture long-term dependencies. They incorporate memory into the network in the form of a cell state, thus allowing for relevant information to be retained for long periods of time. LSTMs are preceded by word embeddings, and these embeddings may be pre-trained or not. In this work, we used pre-trained word embeddings only with the Bi-LSTM model which will be discussed next. We do not use pre-trained word embeddings with the “vanilla” LSTM model. For the vanilla LSTM model, we use the embeddings trained as a part of the Keras [25] embedding layer. We use an LSTM of size 100 (100 hidden units). We use dropout as a regularization mechanism to try to combat overfitting. Finally, we use a sigmoid layer to make the predictions.

Input	Embedding	Dropout	LSTM	Dropout	Dense
In: 50	In: 50	In: 50,100	In: 50,100	In: 100	In: 100
Out: 50	Out: 50,100	Out: 50,100	Out: 100	Out: 100	Out: 1

--->

Figure 1: LSTM Architecture

Figure 1 shows the LSTM architecture that we implemented with a detailed description of its various layers and their respective shapes in Keras.

4.2.2 Bidirectional LSTM

Bidirectional LSTMs are an extension of LSTMs. A drawback of Vanilla LSTMs is that they can learn representations only from the previous time steps. This means that when processing a given word, the model only has access to all the words that came before this word. This way, the model might lose out on a lot of valuable and relevant information that might be present after the word that is currently being processed. In order to combat this problem, we need to be able to look ahead of the current word. A Bidirectional LSTM has the ability to use words both preceding and following the word being processed. This should give it an edge over a vanilla LSTM. We use the pre-trained Glove embeddings [26] in tandem with the bidirectional LSTM. We chose a Glove embedding that generates a 300-dimensional vector embedding for every word. The size of this bidirectional LSTM is 150 (150 hidden units). As with the vanilla LSTM, we use dropout for regularization and sigmoid as the final classification layer.

Input	Embedding	Dropout	BiLSTM	Dropout	Dense
In: 50	In: 50	In: 50,300	In: 50,300	In: 300	In: 300
Out: 50	Out: 50,300	Out: 50,300	Out: 300	Out: 300	Out: 1

-->

Figure 2: Bi-LSTM Architecture

Figure 2 shows the Bidirectional LSTM architecture that was implemented with a detailed description of its various layers and their respective shapes in Keras [25].

4.2.3 CNN

CNNs are a type of feed-forward deep neural network that is primarily applied to data in the form of images. They require minimal pre-processing and are shift invariant. They are used in the domains of image and video recognition, recommender systems and, more recently, NLP. They have already achieved human-level performance on image recognition and have recently made the transition to text classification, where they have been shown to work surprisingly well. CNNs work well with images because they preserve the 2D spatial orientation of the input image. In contrast, texts have a 1D orientation, wherein the sequence of the input words matter. So, we use a 1D CNN to be able to capture this orientation in text. Here we utilize a 1D CNN, followed by a dense layer of size 100. We continue to use sigmoid for the final classification.

Input	Embedding	Dropout	Conv1D	GlobalMax Pooling	Dense	Dropout	Dense
In: 50	In: 50	In: 50,100	In: 50,100	In: 48,100	In: 100	In: 100	In: 100
Out: 50	Out: 50,100	Out: 50,100	Out: 48,100	Out: 100	Out: 100	Out: 100	Out: 1

-->

Figure 3: CNN Architecture

Input	Embedding	Dropout	Conv1D	MaxPooling	LSTM	Dropout	Dense
In: 50	In: 50	In: 50,100	In: 50,100	In: 46,50	In: 11,50	In: 100	In: 100
Out: 50	Out: 50,100	Out: 50,100	Out: 46,50	Out: 11,50	Out: 100	Out: 100	Out: 1

-->

Figure 4: CNN+LSTM Architecture

Figure 3 shows the CNN architecture that is implemented in this paper with a detailed description of its various layers and their respective shapes in Keras [25].

4.2.4 CNN + LSTM

It is interesting to combine the CNN and the LSTM models. LSTM models are very well suited for text classification, but due to their inherent design and sequential processing, they take a long time to train. CNNs, on the other hand, are highly parallel in nature and process the entire data at once. This allows for very short training times, especially when compared to an LSTM. To avoid this shortcoming of LSTMs, we add a convolutional layer before the LSTM layer. This convolutional layer passes a filter over the input text and generates a high-level representation of the input. This high-level representation is then fed to the LSTM instead of the word embeddings. This has a significant positive effect on training times of the LSTM model. Thus we can achieve the best of both worlds by following a CNN with an LSTM, and reducing the training times of the LSTM while retaining its sequence-processing abilities. In this paper, we use a 1D CNN followed by an LSTM of size 100 (100 hidden units). As is the case with all other models, we stick with dropout for regularization and sigmoid for the final classification layer.

Figure 4 shows the CNN+LSTM architecture that is implemented in this paper with a detailed description of its various layers and their respective shapes in Keras [25].

4.2.5 Tools: Artificial Neural Network

Keras was the deep learning framework of choice that was used to implement the various neural network models. Scikit-learn's classification report was used to generate metrics including precision, recall, and F1-Score. The inbuilt tokenizer from Keras was used to tokenize the peer reviews before feeding them to the neural network models.

For each of these networks, a series of hyper-parameters are isolated and tuned, which include parameters such as input batch size, number of memory states, recurrent dropout rate, dropout rate between layers, padding rules, CNN window size and stride size, activation methods, number of epochs to train. Based on classification accuracy on the validation dataset, we have tuned the network into their respective final stages and have achieved results described in the next section.

5. RESULTS

Table 2 displays the total number of suggestions present in the peer review dataset used for this experiment, along with a summary of the weighted average F1-scores. Dataset reformatting was accomplished prior to this so that the number of suggestions to non-suggestions resulted in a mostly equal proportion within the dataset. Abbreviations for the tables are as follows: Rule (as itself), NB (Naïve Bayes), SVM (Support Vector Machine), and N1-N4 as the neural network classifiers (LSTM, BiLSTM, CNN, and CNN+LSTM respectively).

From the results obtained by testing on the Peer Review dataset, we found that the LSTM, CNN, and CNN+LSTM neural network architectures outperform the other classifiers used in this experiment. This relates to our discussion in the literature section where the neural network approaches have demonstrated the greatest potential

Table 2: Accuracy of classifiers on peer-review dataset

# of comments		3878
# containing suggestions.		1939
F1 score	Rule	0.80
	NB	0.84
	SVM	0.88
	N1	0.91
	N2	0.93
	N3	0.92
	N4	0.90

for text classification tasks like suggestion detection. The BiLSTM classifier was the second best at detecting suggestions, followed by the support vector machine, rule-based, and naïve Bayes classifiers. Tables 3 and 4 display the extended breakdown of the results of the classifiers.

These results show precision, recall, and F1-score as generated by the classification report function by Scikit-learn. The number of data points that are false and true is noted in the Support column. The rates of classification for both positives and negatives are included in the table as False and True, along with the support for each. Finally, the weighted average was used to demonstrate the classifier's overall effectiveness by combining the metrics for positives and negatives using a weighting based upon the quantity in Support.

The weighted average is calculated by taking the mean of the false and true predictions by their relevance in the dataset. In terms of these results tables, the weighted average is determined by multiplying the metric (precision, recall, or F1-score) of the false predictions by the percentage of these predictions over the whole dataset. The same calculation is made for the true predictions. Then the value calculated by the False row is added to the value calculated for the True row.

The classifiers performed similarly on overall accuracy, with some deviation in the performance for False and True observations. The F1-scores of the classifiers ranged from 80% up to 93%, where the neural network classifiers were close in performance. Rule 3 of the rule-based classifier was removed due to the tagging of some

words as NNS or NNP that were not indicative of a suggestion. This resulted in numerous false positive classifications and therefore a net loss of overall performance. Rule 2 can also have result in a high false-positive classification rate since it does not consider the context of a tagged pair of words within a sentence, although it was the most effective rule at finding suggestions.

6. DISCUSSION AND FUTURE WORK

In this study, we have compared the performance of a few neural network classifiers against a rule-based classifier on suggestion detection towards multiple datasets. We found that neural network classifiers outperformed existing rule-based and traditional machine learning classifiers across the board.

Despite using a small dataset, we were able to obtain F1 scores in the low 90% range. This demonstrates the potential of the neural network classifiers. When large-scale datasets are obtained, classification scores in the upper 90% range will be probable, therefore reaching the high accuracy levels that some sentiment analysis models have obtained.

One of the biggest challenges we've encountered in this study is the insufficient amount of readily labeled data in hand. Most of the times we would need to manually label datasets that we acquired. Larger datasets would help with tuning and training the classifier. They would also give us the opportunity to train on more balanced sets of data, while not excluding too large of a portion of the overall dataset. Larger training samples would help alleviate the concern that the models may not be able to generalize to new input strings that aren't similar enough to the current training set. However, as noted in Section 2 we removed around 7000 duplicate observations from the dataset, indicating that there are commonly occurring review comments that the model would be tuned for. Additional pattern-matching phrases can be added to the rule-based classifier to improve its accuracy, although additional conditions would be required to consider the context of the phrases within a larger body of text. Also, the inclusion of more types of classifiers such as a decision tree can provide an extended comparison of what classifiers perform best for this text classification task.

As discussed in the introduction, the domain of primary interest revolved around peer reviews. Later on, we might find a way to let instructors use the classifier to grade the quality of peer reviews. We would seek feedback on the effectiveness of the classifier, and suggestions for its improvement. This system could also notify the reviewer that a review should have more

Table 3: Breakdown of accuracy of non-neural network classifiers on peer review

Peer review	Precision			Recall			F1 score			Support		
	Rule	NB	SVM	Rule	NB	SVM	Rule	NB	SVM	Rule	NB	SVM
False	0.76	0.87	0.87	0.88	0.81	0.90	0.82	0.84	0.89	1939	205	205
True	0.86	0.80	0.89	0.73	0.87	0.85	0.79	0.83	0.87	1939	183	183
Avg.	0.81	0.84	0.88	0.80	0.84	0.88	0.80	0.84	0.88	3878	388	388

Table 4: Breakdown of accuracy of neural-network classifiers on peer review

Peer review	Precision				Recall				F1 score				Support
	N1	N2	N3	N4	N1	N2	N3	N4	N1	N2	N3	N4	All
False	0.90	0.94	0.90	0.90	0.92	0.93	0.94	0.90	0.91	0.93	0.92	0.90	189
True	0.92	0.93	0.94	0.90	0.90	0.94	0.90	0.90	0.91	0.93	0.92	0.90	199
Avg.	0.91	0.93	0.92	0.90	0.91	0.93	0.92	0.90	0.91	0.93	0.92	0.90	388

suggestions. Furthermore, the reviewer could have the opportunity to identify any suggestions that were missed by the system. This would function as a useful method for generating labeled training data. These missed suggestions indicated by the reviewer could enable analysis in the form of determining what types of sentences are not being properly detected as suggestions. This system could be deployed as a supplement to instructor grading until enough data has been obtained to train a sufficiently accurate classifier for automatic suggestion extraction.

7. CONCLUSION

In this paper, we have demonstrated several models that parse and classify suggestions. The greater performance of neural network architectures over rule-based methods in this task demonstrates the advantage of classifiers that train on text in a certain domain, rather than following strict rules for classification. Furthermore, forming an extensive list of phrases and keywords that improve a rule-based classifier's performance in a domain is more time consuming than training a statistical classifier. While rule-based approaches have the advantage of being simpler to implement and not requiring data to train on, they are typically outperformed by statistical classifiers provided they have access to enough labeled data.

8. ACKNOWLEDGMENTS

The authors acknowledge the support of the National Science Foundation, through grant #1432347.

9. REFERENCES

- [1] Topping, K.J. 2009. Peer Assessment. Theory into practice. 48, 1 (Jan. 2009), 20–27.
- [2] Nelson, M.M. and Schunn, C.D. 2009. The nature of feedback: how different types of peer feedback affect writing performance. *Instructional Science*. 37, 4 (Jul. 2009), 375–401.
- [3] Demiraslan Çevik, Y. et al. 2015. The effect of peer assessment on problem solving skills of prospective teachers supported by online learning activities. *Studies in Educational Evaluation*. 44, (Mar. 2015), 23–35.
- [4] Liu, X. and Li, L. 2014. Assessment training effects on student assessment skills and task performance in a technology-facilitated peer assessment. *Assessment & Evaluation in Higher Education*. 39, 3 (Apr. 2014), 275–292.
- [5] Lundstrom, K. and Baker, W. 2009. To give is better than to receive: The benefits of peer review to the reviewer's own writing. *Journal of Second Language Writing*. 18, 1 (Mar. 2009), 30–43.
- [6] moocs peer to peer:
https://docs.google.com/presentation/d/1wkJOFgH0DYihsfH_tddcYeI5DMNI9uP1qYgHMPPrf_UY/edit. Accessed: 2018-10-01.
- [7] Van Popta, E. et al. 2017. Exploring the value of peer feedback in online learning for the provider. *Educational Research Review*. 20, (Feb. 2017), 24–34.
- [8] Tsivitanidou, O.E. and Constantinou, C.P. 2016. A study of students' heuristics and strategy patterns in web-based reciprocal peer assessment for science learning. *The Internet and Higher Education*. 29, (Apr. 2016), 12–22.
- [9] Ramachandran, L. et al. 2017. Automated Assessment of the Quality of Peer Reviews using Natural Language Processing Techniques. *International Journal of Artificial Intelligence in Education*. 27, 3 (Sep. 2017), 534–581.
- [10] Xiong, W., & Litman, D. 2011. Automatically predicting peer-review helpfulness. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2* (pp. 502-507). Association for Computational Linguistics.
- [11] Negi, S. and Buitelaar, P. 2017. Chapter 8 - Suggestion Mining From Opinionated Text. *Sentiment Analysis in Social Networks*. F.A. Pozzi et al., eds. Morgan Kaufmann. 129–139.
- [12] Austin, J. L. 1962. *How to do things with words*. Cambridge: Harvard University Press.
- [13] Bird, S. et al. 2009. *Natural Language Processing with Python*. O'Reilly Media, Inc.
- [14] Brun, C. and Hagège, C. 2013. Suggestion Mining: Detecting Suggestions for Improvement in Users' Comments. *Research in Computing Science*. 70, 79.7179 (2013), 5379–5362.
- [15] Marcus, M.P. et al. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*. (1993).
- [16] Murphy, K.P. 2012. *Machine Learning : A Probabilistic Perspective*. MIT Press.
- [17] Gottipati, S. et al. 2018. Text analytics approach to extract course improvement suggestions from students' feedback. *Research and Practice in Technology Enhanced Learning*. 13, 1 (Jun. 2018), 6.
- [18] Gehringer, E. et al. 2007. "Reusable learning objects through peer review: The Expertiza approach," *Innovate—Journal of Online Education* 3:6
- [19] Gehringer, E. et al. 2006. "Expertiza: Reusable learning objects and active learning for distance education," *Proceedings of the UNC Teaching and Learning with Technology conference*, Raleigh
- [20] Krippendorff, Klaus. "Computing Krippendorff's alpha-reliability." (2011).
- [21] Menardi, G. and Torelli, N. 2014. Training and assessing classification rules with imbalanced data. *Data mining and knowledge discovery*. 28, 1 (Jan. 2014), 92–122.
- [22] Kubat, M. et al. 1997. Learning when negative examples abound. *Machine Learning: ECML-97*. M. Someren and G. Widmer, eds. Springer Berlin Heidelberg. 146–153.
- [23] Pedregosa, F. et al. 2011 "Scikit-learn: Machine learning in Python." *Journal of machine learning research*: 2825-2830.
- [24] Wes McKinney. *Data Structures for Statistical Computing in Python*, *Proceedings of the 9th Python in Science Conference*, 51-56 (2010)
- [25] François Chollet. Keras: Deep learning library for theano and tensorflow. <https://github.com/keras-team/keras>, 2015.
- [26] Pennington, J. et al. 2014. "Glove: Global vectors for word representation." *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*

Towards a General Purpose Anomaly Detection Method to Identify Cheaters in Massive Open Online Courses

Giora Alexandron¹, José A. Ruipérez-Valiente² and David E. Pritchard²

¹ Weizmann Institute of Science, Herzl St 234, Rehovot, Israel

² Massachusetts Institute of Technology, 77 Massachusetts Ave, Cambridge (MA), 02139, USA
giora.alexandron@weizmann.ac.il, jruipere@mit.edu, dpritch@mit.edu

ABSTRACT

We propose a general-purpose method for detecting cheating in Massive Open Online Courses (MOOCs) using an Anomaly Detection technique. Using features that are based on measures of aberrant behavior, we show that a classifier that is trained on data of one type of cheating (Copying Using Multiple Accounts) can detect users who perform another type of cheating (unauthorized collaboration). The study exploits the fact that we have dedicated algorithms for detecting these two methods of cheating, which are used as reference models. The contribution of this paper is twofold. First, we demonstrate that a detection method that is based on anomaly detection, which is trained on a known set of cheaters, can generalize to detect cheaters who use other methods. Second, we propose a new time-based person-fit aberrant behavior statistic.

Keywords

MOOCs; Learning Analytics; Anomaly Detection; Cheating

1. INTRODUCTION

Academic dishonesty is one of the endemic problems of higher education. Some studies have reported that up to 95% of college students are engaged in some kind of dishonest behavior [10, 4, 9]. The anonymity of online environments makes it easier for students to cheat [8]. In addition, online learning environments tend to be more heterogeneous, and students might differ significantly in their perception of what constitutes cheating [3]. These might lead students to look for ways to exploit the properties of a learning environment to gain credit without learning the contents [2, 3].

Within the context of online learning, Massive Open Online Courses (MOOCs) have greatly garnered the attention of the media and researchers over the last decade [5]. Estimates are that during 2018 there were more than 100M MOOC learners, more than 11k MOOCs, and more than 900 institutes involved [18]. These large numbers tell some of the story of how MOOCs change the educational landscape.

Giora Alexandron, José A. Ruipérez-Valiente and David Pritchard "Towards a General Purpose Anomaly Detection Method to Identify Cheaters in Massive Open Online Courses" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 480 - 483

MOOCs offer certificates that do not have formal academic status, but are still perceived as valuable, for example in the labor market. Thus, it is not surprising that several studies reported on cheating in MOOCs, for example by plagiarizing peer-review assignments [7]. One of the main cheating methods that was discovered in MOOCs is Copying Using Multiple Accounts (CUMA). Several studies have been successful at detecting CUMA by implementing probabilistic algorithms [13], heuristics [2], and machine learning [17]. A different cheating method, unauthorized collaboration, was reported in [16], and detected using a method that is based on proximity of submissions in time. We note that CUMA and unauthorized collaboration are strictly forbidden by all the major MOOC platforms, e.g., edX¹.

With major MOOC providers currently pivoting towards the direction of professional development and online degrees [15], there is an even greater need for robust techniques to prevent and detect cheating, which can generalize and scale across platforms, topics and courses. The goal of this study is to develop general detection techniques that can identify cheating without assuming a specific pattern. The rationale is to rely on behavioral patterns that capture various types of aberrant behavior, rather than relying on temporal patterns that are specific to a certain method of cheating.

In previous work [1] we *hypothesized* that anomaly detection can be used to build such a general purpose classifier, which 'bootstrap' from one type of cheating to detect other types of cheating. However we were unable to demonstrate this, due to lack of a reference model. The current study extends [1], and demonstrates that a general-purpose cheater detector that is based on anomaly detection can be trained on one type of cheating and then be used to discover other type of cheating. The detector is based on measures of aberrant behavior, such as Guttman Error [11]. As an additional contribution, we also formalize a new time-based aberrant behavior person-fit statistic that was proved useful in discriminating cheaters.

2. METHODOLOGY

2.1 Procedure

The overall rationale of this research is to develop a 'bootstrap' process in which a detector that is trained on one type of cheating is used to build a more general classifier that can detect other types of cheating as well. In our case, the first

¹<https://www.edx.org/edx-terms-service>

type is CUMA, and the other type of cheating is unauthorized collaboration, for which we also have a detector that serves as a reference model.

To test this approach, we use the following cross-validation procedure (with $K=3$, repeated 500 times). First, we train a classifier on a test set, under the assumption that we know to detect only CUMA. In practice, this means that in the test set only CUMA users appear as positive examples (though it may include collaborators data, depending on the random assignment to training/held-out datasets). Second, we use this classifier to classify the held-out dataset. On this set, we check the recall with respect to collaborators. That is, we compute how many of the collaborators in the test set were classified as ‘cheaters’ by the algorithm, and compare it to the fraction of non-cheaters that were identified as cheaters.

In addition, we evaluate the performance of a classifier that is built on a dataset in which both types of cheaters are tagged as positive examples. The rationale for this evaluation is to support future research, in which we intend to check the generalizability of this classifier not only from one method of cheating to the other, but also between MOOCs.

2.2 Data

We use data from an Introductory Physics MOOC offered by the third author through edX on summer 2014. The course consists of 12 required and 2 optional weekly units. A typical unit contains three sections: Instructional e-text/video pages (with interspersed concept questions, aka checkpoints), Homework, and Quiz. Altogether the course contains 273 e-text pages, 69 videos, and ~ 1000 problems. About 13500 students registered to the course, and from them, 502 earned a certificate. This research use the data of 495 certificate earners (7 were omitted due to technical reasons).

2.3 Detecting Cheaters

We define as cheaters those users who use methods that break the code of honor (such as creating multiple accounts or sharing responses with peers) to achieve credit in a way that does not rely on learning. We have algorithms that can detect two specific types of cheating – CUMA, and unauthorized collaboration.

2.3.1 Copying Using Multiple Accounts (CUMA):

To detect CUMA users, we use the algorithm of [2]. It detects 65 users ($\sim 13\%$ of the certificate earners).

2.3.2 Collaborators:

To detect collaborators, we use the algorithm of [16]. Overall, it detects 20 of the certificate earners. However, among those learners, 11 were also classified as CUMA users by the previous algorithm. In these cases, we decided to give priority to the CUMA algorithm, as it represents a more specific behavioral pattern. Hereafter, we refer as ‘collaborators’ to the 9 accounts who were not CUMA users.

2.4 Feature Engineering

We use the following features, divided into three groups:

Video use:

The rationale for this set of features is that cheaters tend

to spend less time on learning resources [1]. As videos are the main learning resource in most MOOCs, this feature can generalize between courses.

i. Watching time: (Log of) The total amount of time, in seconds, that the user spent watching videos.

ii. Fraction of videos watched: The fraction of videos watched. A video is considered as ‘watched’ if the user played more than 30 seconds of it.

Students Performance:

iii. Correct on first attempt: The fraction of the items that were solved correctly on first attempt.

iv. Mean time to correct: The average time on task, for items solved correctly.

v. Fraction of correct-in-less-than-30 seconds: The fraction of the items that were solved correctly in less than 30 seconds. The 30-second threshold is taken from [14].

Person-fit statistics:

vi. Guttman Error (GE): The number of item pairs in which an easier item is answered incorrectly and a more difficult item is answered correctly, normalized by the total number of pairs [11]. To make our method more general, we use the non-parametric variant, as parametric models (e.g. 2PL IRT) are difficult to fit on MOOCs data. It is computed in *R* using the package *PerFit* [19]

vii. Guttman Error on time-on-task (GE-time): This is a new aberrant behavior person-fit statistic that we propose. It basically applies the notion of Guttman Error to time-on-task. It is described in more detail in the Appendix.

2.4.1 Z-scores and Feature Selection

The independent variables were standardized using *z*-scores, to enable comparing the relative importance of features based on standardized logistic regression coefficients [12], and to allow (in the future) generalizing to other MOOCs. For feature selection, we use a LASSO logistic regression and pick the features that have a non-zero coefficient (the tuning parameter λ is chosen via cross-validation). This is done in *R* using the package *glmnet* [6].

3. RESULTS

This section is organized as follows. First, we report on the results of the feature selection. Second, we present the distribution of the features among CUMA, collaborators, and non-cheaters. Third, we present the performance of the classifier trained on the CUMA users, when used to detect collaborators. Fourth, we report on the performance of a classifier that is trained on both type of cheating.

3.1 Feature Selection

The features with non-zero value are GE, GE-time, fraction of videos watched, and fraction of questions answered in less than 30 seconds.

3.2 Group Differences

Figure 1 presents the differences between the three groups – CUMA users (red), collaborators (black), and non-cheaters (blue), with respect to the four features.

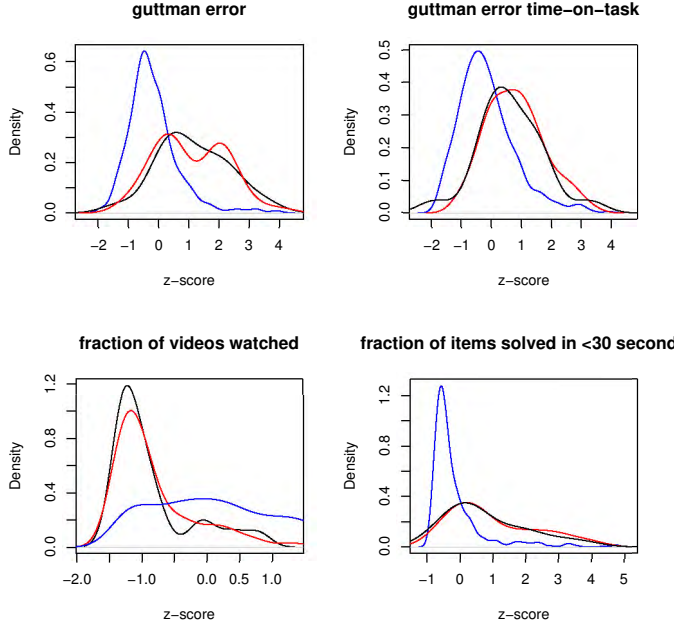


Figure 1: Distribution of the independent variables among CUMA users (red), collaborators (black), and non-cheaters (blue).

3.3 Detecting unauthorized collaboration with a classifier build on CUMA

As described in Section 2, we train a logistic regression model that receives a training set with only ‘CUMA’ users tagged as positive examples, and use it to detect ‘collaborators’. To ensure that we accurately simulate a scenario in which no information on collaborators exists during the training phase, the training set is used for fitting the model and tuning hyper-parameters, and for model-selection. Collaborators might exist in the training data (depending on the random assignment to training/test), but as negative examples (i.e., non-cheaters).

The results ($recall = \frac{TP}{TP+FN}$) of applying this cross-validation process with $K = 3$, repeated over 500 times, are presented in Figure 2. Overall, $mean(recall) = 0.72$, $sd = 0.12$.

For negative examples (neither CUMA nor collaborators), the mean amount of miss-classification ($\frac{FP}{FP+TN}$) is 0.16 ($sd = 0.01$). This means that a collaborator is 4.5 times more likely to be classified as ‘positive’, than a non-cheater.

3.4 Building a general classifier

Next, we turn to build a classifier on a dataset that contains both types of cheaters as positive examples. The rationale that underlies this is to build a classifier that can 1) ‘bootstrap’: use data that includes two types of cheating to discover additional ones; and 2) build a global classifier that can (hopefully) generalize across MOOCs.

1. **Feature Selection.** This yields the same set of features as reported above.

2. **Performance of the classifier.** We evaluate the per-

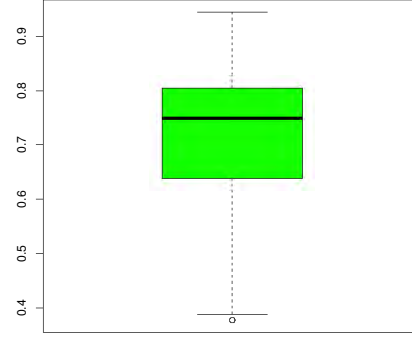


Figure 2: Fraction of collaborators identified by the classifier.

Table 1: Confusion matrix.

		Predicted	
		False	True
Actual	False	377	45
	True	22	51

formance of the classifier using cross-validation and by observing the in-sample classification error. For cross-validation, we measure the AUC of 500 5-fold cross-validation runs. The results are presented in Figure 3. $mean(auC) = 0.85$, $sd = 0.01$.

The confusion matrix for in-sample classification is given in Table 1. The classifier identifies 45 additional users as ‘cheaters’. Applying the previous results, we can hypothesize that among these, ~ 35 are ‘real’ cheaters who are not detected by our previous algorithms, and that ~ 10 are true false positives.

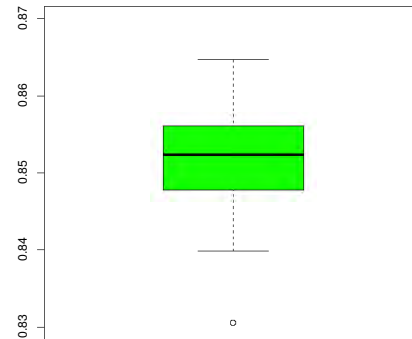


Figure 3: AUC of the classifier

4. DISCUSSION

In previous work [1] we *hypothesized* that anomaly detection can be used to build classifiers that can generalize from

known cheating methods to *unknown* ones. However, in our previous work we were unable to provide an empirical evidence for this, due to lack of reference model on a second type of cheating.

The current paper re-visits this approach, exploiting the fact that we now have a dedicated algorithm for detecting unauthorized collaboration, which serves as a reference model. Based on this, we demonstrate that an anomaly-detection based classifier can generalize from one type of cheating to another with high accuracy.

The classifier uses 4 aberrant behavior features. One of them is a new time-based aberrant behavior person-fit statistics that we propose, which was found to be very effective in discriminating cheaters. We name it *Guttman Error-time*.

The power of our approach lies in the fact that 1) it does not rely on prior assumptions on the cheating method, and thus does not require dedicated algorithms that are tailored to a specific method; and 2) the features that it uses are relatively simple to compute, and do not rely on fitting sophisticated parametric models (e.g., IRT). This makes our method scalable and easy to implement across contexts.

Future research. In the future we intend to study whether this method can generalize not only between different methods within the same course, but also between courses.

5. ACKNOWLEDGMENTS

GA's research is supported by the Israeli Ministry of Science and Technology under project no. 713257.

6. REFERENCES

- [1] G. Alexandron, S. Lee, Z. Chen, and D. E. Pritchard. Detecting cheaters in moocs using item response theory and learning analytics. In *UMAP'16*, 2016.
- [2] G. Alexandron, J. A. Ruipérez-Valiente, Z. Chen, P. J. Muñoz-Merino, and D. E. Pritchard. Copying@Scale: Using harvesting accounts for collecting correct answers in a MOOC. *Computers & Education*, 108:96–114, 2017.
- [3] E. W. Black, J. Greaser, and K. Dawson. Academic dishonesty in traditional and online classrooms: Does the. *Journal of asynchronous learning networks*, 12:23–30, 2008.
- [4] S. Davis. Academic dishonesty in the 1990s. *The Public Perspective*, 1993.
- [5] S. I. De Freitas, J. Morgan, and D. Gibson. Will moocs transform learning and teaching in higher education? engagement and course retention in online learning provision. *British Journal of Educational Technology*, 46(3):455–471, 2015.
- [6] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *J. of Statistical Software*, 33(1):1–22, 2010.
- [7] L. Gibbs. Yes, plagiarism: How sad is that. *Coursera Fantasy: Blogging My Way through a MOOC*, 2012.
- [8] C. G. King, R. W. Guyette Jr, and C. Piotrowski. Online exams and cheating: An empirical analysis of business students' views. *J. of Educators Online*, 2009.
- [9] D. L. McCabe and L. K. Trevino. Academic dishonesty: Honor codes and other contextual influences. *J. of Higher Education*, 1993.
- [10] D. L. McCabe, L. K. Trevino, and K. D. Butterfield. Cheating in academic institutions: A decade of research. *Ethics & Behavior*, 11(3):219–232, 2001.
- [11] R. R. Meijer. The Number of Guttman Errors as a Simple and Powerful Person-Fit statistic. *Applied Psychological Measurement*, 18(4), 1994.
- [12] S. Menard. Six approaches to calculating standardized logistic regression coefficients. *The American Statistician*, 58(3):218–223, 2004.
- [13] C. G. Northcutt, A. D. Ho, and I. L. Chuang. Detecting and preventing “Multiple-Account” cheating in massive open online courses. *Computers & Education*, 100:71–80, 2016.
- [14] D. J. Palazzo, Y.-J. Lee, R. Warnakulasooriya, and D. E. Pritchard. Patterns, correlates, and reduction of homework copying. *Phys. Rev. ST Phys. Educ. Res.*, 6, 2010.
- [15] J. Reich and J. A. Ruipérez-Valiente. The MOOC pivot. *Science*, 363(6423):130–131, 2019.
- [16] J. A. Ruipérez-Valiente, S. Joksimović, V. Kovanović, D. Gašević, P. J. Muñoz-Merino, and C. Delgado Kloos. A data-driven method for the detection of close submitters in online learning environments. In *Proceedings of WWW'17 Companion*, pages 361–368, 2017.
- [17] J. A. Ruipérez-Valiente, P. J. Muñoz-Merino, G. Alexandron, and D. E. Pritchard. Using machine learning to detect ‘multiple-account’ cheating and analyze the influence of student and problem features. *IEEE Transactions on Learning Technologies*, 2017.
- [18] D. Shah. By The Numbers: MOOCs in 2018. Class Central, 2019.
- [19] J. N. Tendeiro, R. R. Meijer, and A. S. M. Niessen. PerFit: An R package for person-fit analysis in IRT. *Journal of Statistical Software*, 74(5):1–27, 2016.

APPENDIX: Guttman Error-time – a new time-based person-fit statistic

Guttman Error-time applies the idea of *Guttman Error* to time-on-task. Let us define the following notations: $mean_ttc(u)$ = the mean time-to-correct of user u on all the items u solved correctly.

$ttc(u, i)$ = time-to-correct of user u on item i .

Now assume we have the Time-To-Correct matrix TTC with $TTC[u, i] = ttc(u, i)$

Let us build a new matrix Boolean-TTC, such that:

Boolean-TTC[u,i] = 0 if $ttc(u, i) > mean_ttc(u)$, 1 otherwise.

This means that an item on which many students were slower than usual, will have a lot 0's in its column. Intuitively, this is the equivalence of a ‘hard’ item in the correct-on-first-attempt matrix. Now, we define:

$GE - time = GE_{normed}(Boolean - TTC)$

A Novel Use of Educational Data Mining to Inform Effective Management of Academic Programs

Anwar Ali Yahya
Najran University
Najran, Saudi Arabia
Thamar University
Thamar, Yemen
aaesmail@nu.edu.sa

Fekri Abdulwadod Mohammed
Najran University
Najran, Saudi Arabia
faahmed@nu.edu.sa

Addin Osman
Najran University
Najran, Saudi Arabia
aomaddin@nu.edu.sa

ABSTRACT

In outcome-based academic programs, Program Educational Objectives (PEOs) and Student Outcomes (SOs) are two cores, which are mapped to each other in a hierarchical manner. This paper presents a novel application of educational data mining to discover useful knowledge on these components. More specifically, it applies association rule mining techniques to discover mapping rules between PEOs and SOs and correlation rules among SOs. As a case study, this paper demonstrates how association rule mining techniques is applied to discover PEO-SOs mapping rules and SOs correlation rules in engineering programs. To this end, a set of 152 self-study reports of engineering programs, accredited by American Board for Engineering and Technology-Engineering Accreditation Commission (ABET-EAC), have been collected and the mapping data between their PEOs and ABET-EAC SOs have been extracted. The dataset has been processed and transformed into a representation suitable for association rule mining techniques. This involves identifying a set of PEOs labels, annotating data instances with PEOs labels, and projecting each multi-label data instance into a set of single-label instances. Apriori algorithm is then applied to generate rules involving PEOs and SOs. The generated rules are then filtered to obtain mapping rules between PEOs and ABET-EAC SOs and the correlation rules among SOs. The filtered rules are then used to draw a set of generic rules for mapping each PEO to ABET-EAC SOs and to figure out potential correlations exist among ABET-EAC SOs.

Keywords

Association rule mining, Educational data mining, Academic program management, Program educational objectives, Student outcomes; ABET accreditation.

1. INTRODUCTION

Outcome Based Education (OBE) [1] is a new educational system that focuses on graduate attributes which allow students to accept the challenges, adopt to technological changes, and translate their knowledge to new contexts for the benefit of the society. In this sense, OBE-based academic programs are designed to develop various abilities as per the requirements of graduate attributes. An

essential step in designing OBE academic program is the identification of long-term PEOs and SOs [2]. While SOs represent the knowledge, skills, and capabilities that students should possess by the time of graduation, PEOs represent the achievements graduates should attain few years (3 to 5 years and more) after graduation. Having specified the PEOs and SOs of an OBE academic program and established the mapping between them, the curriculum, teaching and learning strategies, and assessment strategies are then designed in such a way that students ultimately gain knowledge and develop skills stated in the SOs. In this manner, the curriculum is viewed as a set of courses aim to attain certain Course Outcomes (COs) that map to SOs which themselves map to PEOs which in turn map to the mission of the institution in a hierarchical structure shown in Figure 1.

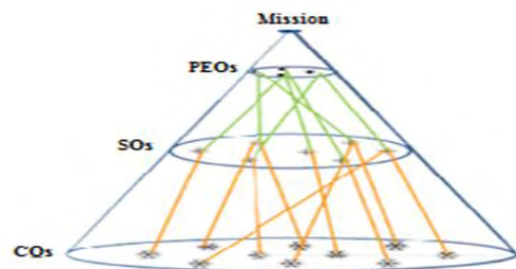


Figure 1. Structure of OBE academic program [1]

Despite the crucial role of PEOs-SOs mapping, there is a consensus on the lingering confusion, among practitioners, related to them and how they are correlated to each other [3]. The drastic consequences of such confusion is a poor design of curriculum and teaching strategies and a misleading assessment of PEOs and ultimately inaccurate corrective plan. To this end, this paper proposes using Association Rules Mining (ARM) techniques [4] to inductively interrogate a set of PEOs-SOs mapping data. More specifically, this paper demonstrates an application of Apriori algorithm to a set of PEOs-SOs mapping data extracted from self-study reports (SSRs) of a number of engineering programs accredited by ABET-EAC. This results in a huge sets of rules which are further filtered to arrive at specific knowledge in a form of generic rules for mapping PEOs to ABET-EAC SOs and correlations rules among ABET-EAC SOs.

2. METHODOLOGY

The methodology of this work has been customized from the general methodology of the knowledge discovery process depicted in Fig. 2. It involves raw data collection, selection, preprocessing, transformation, mining, and evaluation [4]. In the data collection step, a raw data is collected and used to create a target dataset or focusing on a subset of variables or data samples on which discovery is to be performed. The target data is cleaned

Anwar Ali Yahya, Fekri Mohammed and Addin Osman "A Novel Use of Educational Data Mining to Inform Effective Management of Academic Programs" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 484 - 487

and preprocessed in order to obtain consistent data in the pre-processing step. The transformation step transforms the data using dimensionality reduction or other transformation methods. The step of data mining applies procedures to search for patterns of interest in a particular representational form, depending on the data mining objective. Finally, in the interpretation/evaluation step, the mined patterns are interpreted and evaluated.

A particular data mining techniques, which are applied to discover all hidden associations that satisfy some user-predefined criteria are association rules algorithms [5]. A widely used algorithm for the association rules mining is the Apriori algorithm [6]. It is based on the following rule: All sub-itemsets of a frequent itemset must also be frequent. Using this rule, Apriori algorithm prunes a huge amount of itemsets examinations since it is certain that they are not frequent. Frequent sub-itemsets are extended one item at a time (candidate generation), and groups of candidates are examined. It terminates when no further extensions are found.

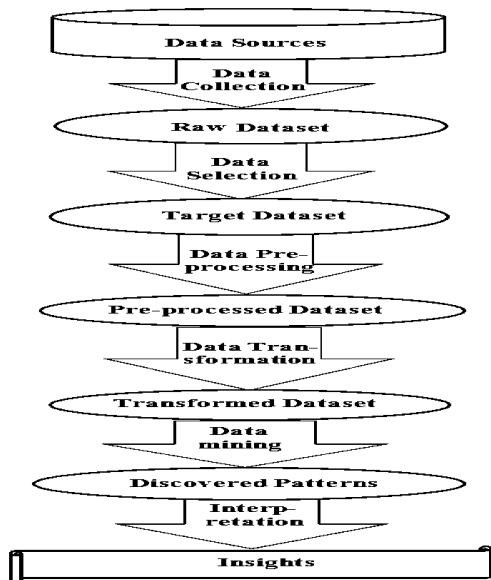


Figure 2. Knowledge Discovery Methodology

3. ARM OF PEOs-SOs DATA IN ENGINEERING PROGRAMS

3.1 Raw Data Collection

The raw data is a set of SSRs of 152 engineering programs [7] accredited by ABET-EAC between 2000 and 2017 and distributed over the years as in Figure 3.

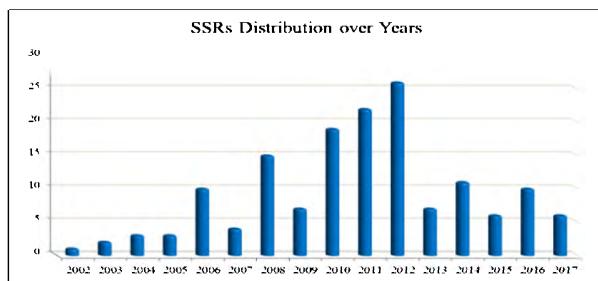


Figure 3. SSRs distribution over years

3.2 Data Selection

The PEOs-SOs mapping data are extracted from sub-section B of the third criteria (Student Outcomes) of each SSR and

consolidated in a table form as shown in Table 1, where symbol ✓ (×) indicates the presence (absence) of the SO in the mapping with the PEO.

Table 1. Excerpts from PEOs-SOs mapping data

PEOs	a	b	c	d	e	f	g	h	i	j	k
Practice the disciplines of transportation, environmental, structural, water resources, and geotechnical engineering, and/or related fields.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Engage in advanced education, research, and development	✓	✓	✓	×	✓	×	×	×	✓	×	✓
Pursue continuing education and professional licensure	×	×	×	×	×	×	×	×	✓	×	✓
Act in a responsible, professional and ethical manner	×	×	×	×	×	×	×	✓	×	✓	×

3.3 Data Preprocessing

The preprocessing of the dataset involves substantial verification and validation of the content, attempts to remove spurious or duplicated objectives, fulfilling the objectives and outcomes format.

3.4 Data Transformation

A transformation procedure is proposed to transform each PEO into a set of labels representing attributes expressed in its text as follows.

3.4.1 PEOs Labels Set Identification

PEOs are grouped into a finite set of common attributes. Based on PEOs wordings of a number of engineering programs, a set of common PEOs labels has been proposed as shows in Table 2.

Table 2. PEOs Labels Set

No	PEOs Category	PEO Label	Frequency
1	Life-long Learning	LL	130
2	Communication	C	82
3	Leadership	L	62
4	Teaming	T	98
5	Ethical Conduct	EC	81
6	Professionalism	P	137
7	Social and Community	SC	78
8	Career Success	CS	127
9	Technical Competency	TC	212
10	Knowledge	KC	96
11	Graduate Studies	GS	64
12	Others	O	29

3.4.2 PEO Annotation

Each instance of PEOs-SOs mapping data is annotated with one or more labels that match its PEO text. Three annotators initially annotated the dataset individually. The three annotators then met to resolve the conflicting cases. Table 3 shows excerpts of the dataset after PEOs annotation. The symbol ✓(×) denotes the inclusion (exclusion) of a particular SO in the mapping.

Table 3. PEOs Labeled Dataset

PEOs Labels	a	b	c	d	e	f	g	h	i	j	K
TC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LL GS	✓	✓	✓	×	✓	×	×	×	✓	×	✓
LL P	×	×	×	×	×	×	×	×	✓	×	✓

3.4.3 Data Projection

Each multi-label data instance is projected into a set of single-label data instances by making a single copy for each PEO appears in the original data instance. This results in an enlarged dataset with 1196 single-label data instances. Table 4 shows the excerpts of Table 3 projection.

Table 4. Projected PEOs Labeled Dataset

PEOs Labels	a	b	c	d	e	f	g	h	i	j	k
TC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LL	✓	✓	✓	×	✓	×	×	×	✓	×	✓
GS	✓	✓	✓	×	✓	×	×	×	✓	×	✓
LL	×	×	×	×	×	×	×	×	✓	×	✓
P	×	×	×	×	×	×	×	×	✓	×	✓

3.5 Association Rules Mining Application

Apriori algorithm is applied under WEKA framework [8]. It works by iteratively generating the frequent k-item sets whose their count equal to or greater than a pre-specified minimum support count, which is calculated as in Eq. 1

$$support(PEO \Rightarrow SOs) = \frac{count(PEO \cup SOs)}{N} \quad (1)$$

Then in the rules generation, rules that satisfy the user specified confidence threshold is generated. The confidence level of the rules is calculated using Eq.2

$$confidence(PEO \Rightarrow SOs) = \frac{count(PEO \cup SOs)}{count(PEO)} \quad (2)$$

In this research the minimum support and minimum confidence are set to low values, 0.1, to generate as many rules as possible.

3.6 Evaluation

The application of Apriori algorithm results in a huge number of rules, therefore, the generated rules are then filtered and interpreted to extract useful insights.

3.6.1 PEOs-SOs Mapping Rules

The generated rules are filtered to extract PEOs-SOs mapping rules. These rules are characterized by having a particular PEO in its antecedent and a combination of SOs in its consequent. The filtered rules are then sorted based on their confidence and the top-10 rules are presented in Table 5.

Table 5: Top-10 PEOs-SOs mapping rules

Life-Long Learning	Communication	Leadership	Teaming	Ethical Conduct	Professionalism	Social & Community	Career Success	Technical Competency	Knowledge Competency	Graduate Study
$LL \xrightarrow{0.85} i$	$C \xrightarrow{0.90} g$	$L \xrightarrow{0.76} d$	$T \xrightarrow{0.85} d$	$EC \xrightarrow{0.89} \bar{b}$	$P \xrightarrow{0.80} \bar{b}$	$SC \xrightarrow{0.90} \bar{b}$	$CS \xrightarrow{0.74} e$	$TC \xrightarrow{0.74} a$	$KC \xrightarrow{0.75} \bar{d}$	$GS \xrightarrow{0.77} i$
$LL \xrightarrow{0.73} \bar{d}$	$C \xrightarrow{0.80} \bar{i}$	$L \xrightarrow{0.71} \bar{b}$	$T \xrightarrow{0.73} \bar{a}$	$EC \xrightarrow{0.86} \bar{k}$	$P \xrightarrow{0.73} \bar{a}$	$SC \xrightarrow{0.87} \bar{a}$	$CS \xrightarrow{0.71} a$	$TC \xrightarrow{0.71} c$	$KC \xrightarrow{0.71} \bar{g}$	$GS \xrightarrow{0.72} a$
$LL \xrightarrow{0.72} \bar{b}$	$C \xrightarrow{0.82} \bar{b}$	$L \xrightarrow{0.69} \bar{a}$	$T \xrightarrow{0.72} \bar{b}$	$EC \xrightarrow{0.85} \bar{a}$	$P \xrightarrow{0.72} \bar{c}$	$SC \xrightarrow{0.87} \bar{a} \wedge \bar{b}$	$CS \xrightarrow{0.70} k$	$TC \xrightarrow{0.71} e$	$KC \xrightarrow{0.68} \bar{i}$	$GS \xrightarrow{0.67} e$
$LL \xrightarrow{0.70} \bar{e}$	$C \xrightarrow{0.79} \bar{a}$	$L \xrightarrow{0.69} \bar{k}$	$T \xrightarrow{0.7} \bar{a} \wedge \bar{b}$	$EC \xrightarrow{0.83} \bar{e}$	$P \xrightarrow{0.72} \bar{e}$	$SC \xrightarrow{0.85} \bar{k}$	$CS \xrightarrow{0.69} c$	$TC \xrightarrow{0.71} \bar{f}$	$KC \xrightarrow{0.65} \bar{f}$	$GS \xrightarrow{0.66} k$
$LL \xrightarrow{0.68} \bar{c}$	$C \xrightarrow{0.79} g \wedge \bar{i}$	$L \xrightarrow{0.66} g$	$T \xrightarrow{0.68} \bar{i}$	$EC \xrightarrow{0.83} \bar{a} \wedge \bar{b}$	$P \xrightarrow{0.70} \bar{b} \wedge \bar{c}$	$SC \xrightarrow{0.81} \bar{e}$	$CS \xrightarrow{0.69} b$	$TC \xrightarrow{0.71} \bar{i}$	$KC \xrightarrow{0.64} \bar{d} \wedge \bar{g}$	$GS \xrightarrow{0.63} b$
$LL \xrightarrow{0.68} \bar{g}$	$C \xrightarrow{0.77} \bar{f}$	$L \xrightarrow{0.66} \bar{i}$	$T \xrightarrow{0.68} \bar{c}$	$EC \xrightarrow{0.81} f$	$P \xrightarrow{0.64} \bar{d}$	$SC \xrightarrow{0.81} \bar{b} \wedge \bar{k}$	$CS \xrightarrow{0.66} a \wedge e$	$TC \xrightarrow{0.68} b$	$KC \xrightarrow{0.58} \bar{k}$	$GS \xrightarrow{0.59} a \wedge e$
$LL \xrightarrow{0.65} \bar{a}$	$C \xrightarrow{0.76} \bar{a} \wedge \bar{b}$	$L \xrightarrow{0.66} \bar{a} \wedge \bar{k}$	$T \xrightarrow{0.65} \bar{e}$	$EC \xrightarrow{0.81} \bar{b} \wedge \bar{e}$	$P \xrightarrow{0.64} f$	$SC \xrightarrow{0.79} \bar{a} \wedge \bar{k}$	$CS \xrightarrow{0.65} b \wedge e$	$TC \xrightarrow{0.68} k$	$KC \xrightarrow{0.57} g \wedge \bar{i}$	$GS \xrightarrow{0.58} \bar{d}$
$LL \xrightarrow{0.64} \bar{b} \wedge \bar{c}$	$C \xrightarrow{0.74} \bar{b} \wedge g$	$L \xrightarrow{0.66} \bar{b} \wedge \bar{k}$	$T \xrightarrow{0.64} g$	$EC \xrightarrow{0.81} \bar{b} \wedge \bar{k}$	$P \xrightarrow{0.64} \bar{b} \wedge \bar{b}$	$SC \xrightarrow{0.90} \bar{b} \wedge \bar{k}$	$CS \xrightarrow{0.65} a \wedge b$	$TC \xrightarrow{0.67} j$	$KC \xrightarrow{0.55} \bar{h}$	$GS \xrightarrow{0.56} \bar{f}$
$LL \xrightarrow{0.64} \bar{b} \wedge \bar{e}$	$C \xrightarrow{0.74} \bar{b} \wedge \bar{i}$	$L \xrightarrow{0.65} \bar{c}$	$T \xrightarrow{0.64} \bar{b} \wedge \bar{c}$	$EC \xrightarrow{0.80} \bar{a} \wedge \bar{k}$	$P \xrightarrow{0.64} \bar{a} \wedge \bar{b}$	$SC \xrightarrow{0.90} \bar{a} \wedge \bar{b} \wedge \bar{k}$	$CS \xrightarrow{0.65} a \wedge c$	$TC \xrightarrow{0.65} \bar{g}$	$KC \xrightarrow{0.54} \bar{d} \wedge \bar{i}$	$GS \xrightarrow{0.56} j$
$LL \xrightarrow{0.64} \bar{d} \wedge i$	$C \xrightarrow{0.73} \bar{a} \wedge g$	$L \xrightarrow{0.65} \bar{a} \wedge \bar{b}$	$T \xrightarrow{0.64} \bar{k}$	$EC \xrightarrow{0.78} \bar{a} \wedge \bar{e}$	$P \xrightarrow{0.68} \bar{g}$	$SC \xrightarrow{0.78} \bar{a} \wedge \bar{e}$	$CS \xrightarrow{0.64} b \wedge c$	$TC \xrightarrow{0.65} \bar{h}$	$KC \xrightarrow{0.53} \bar{b}$	$GS \xrightarrow{0.56} a \wedge b$

Using the top-10 rules for each PEO, a more generic representation can be obtained as shown in Table 6. In this Table the symbols ✓, ×, and ? indicate, respectively, the presence, absence, and undetermined state of a given SO in the consequent of the PEO rule. It can be observed that the *Ethical Conduct* and *Social and Community* PEOs (*EC* and *SC*) have the highest average confidence, whereas *Knowledge Competency* and *Graduate Studies* PEO (*KC* and *GS*) have the lowest. Moreover, it can be observed that the *Social and Community* and *Knowledge Competency* PEOs (*SC* and *KC*) are not dependent on the presence of any SOs. They mainly depend on the absence of different combinations of SOs. The *Lifelong Learning*, *Communication*, *Ethical Conduct*, and *Professionalism* PEOs (*LL*, *C*, *EC*, *P*) depend on the presence of a single SO, which indicates a one to one mapping between the graduate attribute of the PEO and the skills of the SO.

Table 6. Recommended PEOs-SOs mapping

PEO	SOs											Conf. Interval	Avg. Conf.
	a	b	c	d	e	f	g	h	i	j	k		
LL	×	×	×	×	×	?	×	?	✓	?	?	0.65-0.85	0.69
C	×	×	?	?	?	×	✓	?	×	?	?	0.77-0.91	0.79
L	×	×	×	✓	?	?	✓	?	×	?	×	0.65-0.76	0.68
T	×	×	×	✓	×	?	✓	?	×	?	×	0.66-0.82	0.73
EC	×	×	?	?	×	✓	?	?	?	?	×	0.81-0.89	0.83
P	×	×	×	×	×	✓	×	?	?	?	?	0.68-0.8	0.71
SC	×	×	?	?	×	?	?	?	?	?	×	0.81-0.9	0.83
CS	✓	✓	✓	?	✓	?	?	?	?	?	✓	0.69-0.74	0.68
TC	✓	✓	✓	?	✓	×	×	×	×	×	✓	0.65-0.74	0.69
KC	?	×	?	×	?	×	×	×	×	?	×	0.53-0.75	0.6
GS	✓	✓	?	×	✓	×	?	?	✓	✓	✓	0.56-0.77	0.6

3.6.2 ABET-EAC SOs Correlations Rules

In order to discover correlations among ABET-EAC SOs, the generated rules are filtered to extract rules of the form $SOx \Rightarrow SOy$ and $\bar{SOx} \Rightarrow \bar{SOy}$, where SOx and SOy are two given SOs. Table 7 shows the extracted rules along with their confidence values. The correlation between SOx and SOy is defined in terms of the confidence in their equivalency as follows

$$Correlation(SOx, SOy) = Confidence(SOx \Leftrightarrow SOy) \quad (3)$$

$$= Confidence((SOx \wedge SOy) \vee (\bar{SOx} \wedge \bar{SOy})) \quad (4)$$

$$= P((SOx \wedge SOy) \vee (\bar{SOx} \wedge \bar{SOy})) \quad (5)$$

$$= P(SOx \wedge SOy) + P(\bar{SOx} \wedge \bar{SOy}) \quad (6)$$

$$= P(SOx | SOy) \times P(SOy) + \bar{SOx} | \bar{SOy} \times P(\bar{SOy}) \quad (7)$$

$$= Confidence(SOx \Rightarrow SOy) \times P(SOy) + Confidence(\bar{SOx} \Rightarrow \bar{SOy}) \times P(\bar{SOy}) \quad (8)$$

By applying Eq. 8 to the data of Table 7, the correlations among ABET-EAC SOs pairs can be calculated and a correlations graph can be depicted as shown in Figure 4, where 0.65 is assumed as a minimum confidence for drawing a link between PEOs pairs. This threshold value allows each PEO to be linked to at least one PEO. In the correlation graph, the thicker the link is, the stronger the correlation between PEOs is.

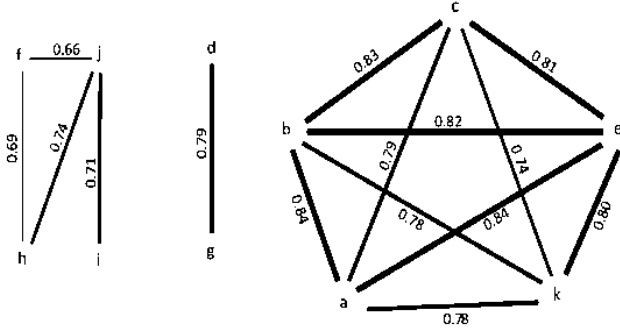


Figure 4. ABET-EAC SOs correlations graph

Obviously, three SOs clusters appear in the correlation graph. The first cluster involves *technical skills* SOs (a, b, c, e, and k). The second clusters involves *process skills* SOs (d and g), and the third cluster involves *awareness skills* SOs (f, h, i, and j).

4. CONCLUSION

Association rules data mining techniques is proposed to discover useful insights on the rules that govern the mapping between PEOs and SOs and the correlations among SOs of academic programs. The discovered insights are useful for academicians to manage academic programs. In addition, a number of interesting correlations between PEOs and ABET-EAC SOs and among ABET-EAC SOs themselves have been discovered.

5. ACKNOWLEDGEMENTS

This work was supported by the Najran University [NU/ESCI/16/022].

6. REFERENCES

- [1] S. Subbaraman, R. R. Jagtap and S. S. Shinde, "Outcome Based Learning: A Case Study," in *IEEE International Conference in MOOC, Innovation and Technology in Education (MITE)*, 2013.
- [2] R. Shivakumar, S. H. Usha, N. A. Chetan, K. Sainath and M. Samita, "Establishing Program Educational Objectives," *Journal of Engineering Education Transformations*, vol. 29, no. 2, pp. 53-58, 2015.
- [3] ABET, San Diego, CA: ABET, 2005.
- [4] J. Han, M. Kamber and J. Pei, "Data Mining: Concepts and Techniques," in *The Morgan Kaufmann Series in Data Management Systems*, Elsevier, 2011.
- [5] S. Ougiaroglou and G. Paschalis, "Association Rules Mining from the educational data of ESOG web-based application," in *IFIP Advances in Information and Communication Technology*, vol. 382, Berlin, Heidelberg, Springer, 2012.
- [6] R. Agrawal and S. Ramakrishnan, "Fast algorithms for mining association rules in large databases," in *the 20th International Conference on Very Large Data Bases, VLDB*, Santiago, Chile pp, 1994.
- [7] A. Osman, A. A. Yahya and M. B. Kamal, "A Benchmark Collection for Mapping Program Educational Objectives to ABET Student Outcomes: Accreditation," in *Advances in Intelligent Systems and Computing*, vol. 753, M. Alenezi and B. Qureshi, Eds., Riyadh, Springer, 2018.
- [8] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, "The WEKA data mining software: an update," vol. 11, no. 1, pp. 10-18, 2009.

Table 7: Confidence ($SOx \Rightarrow SOy$) and Confidence ($\overline{SOx} \Rightarrow \overline{SOy}$)

$\overset{conf}{a} \Rightarrow SOy$ $\overset{conf}{\bar{a}} \Rightarrow \overline{SOy}$	$\overset{conf}{b} \Rightarrow SOy$ $\overset{conf}{\bar{b}} \Rightarrow \overline{SOy}$	$\overset{conf}{c} \Rightarrow SOy$ $\overset{conf}{\bar{c}} \Rightarrow \overline{SOy}$	$\overset{conf}{d} \Rightarrow SOy$ $\overset{conf}{\bar{d}} \Rightarrow \overline{SOy}$	$\overset{conf}{e} \Rightarrow SOy$ $\overset{conf}{\bar{e}} \Rightarrow \overline{SOy}$	$\overset{conf}{f} \Rightarrow SOy$ $\overset{conf}{\bar{f}} \Rightarrow \overline{SOy}$	$\overset{conf}{g} \Rightarrow SOy$ $\overset{conf}{\bar{g}} \Rightarrow \overline{SOy}$	$\overset{conf}{h} \Rightarrow SOy$ $\overset{conf}{\bar{h}} \Rightarrow \overline{SOy}$	$\overset{conf}{i} \Rightarrow SOy$ $\overset{conf}{\bar{i}} \Rightarrow \overline{SOy}$	$\overset{conf}{j} \Rightarrow SOy$ $\overset{conf}{\bar{j}} \Rightarrow \overline{SOy}$	$\overset{conf}{k} \Rightarrow SOy$ $\overset{conf}{\bar{k}} \Rightarrow \overline{SOy}$
$a \Rightarrow b$ $\bar{a} \Rightarrow \bar{b}$	$b \Rightarrow a$ $\bar{b} \Rightarrow \bar{a}$	$c \Rightarrow a$ $\bar{c} \Rightarrow \bar{a}$	$d \Rightarrow a$ $\bar{d} \Rightarrow \bar{a}$	$e \Rightarrow a$ $\bar{e} \Rightarrow \bar{a}$	$f \Rightarrow a$ $\bar{f} \Rightarrow \bar{a}$	$g \Rightarrow a$ $\bar{g} \Rightarrow \bar{a}$	$h \Rightarrow a$ $\bar{h} \Rightarrow \bar{a}$	$i \Rightarrow a$ $\bar{i} \Rightarrow \bar{a}$	$j \Rightarrow a$ $\bar{j} \Rightarrow \bar{a}$	$k \Rightarrow a$ $\bar{k} \Rightarrow \bar{a}$
0.76 0.90	0.85 0.83	0.75 0.74	0.52 0.64	0.80 0.87	0.40 0.55	0.48 0.61	0.41 0.56	0.42 0.56	0.41 0.55	0.74 0.82
$a \Rightarrow c$ $\bar{a} \Rightarrow \bar{c}$	$b \Rightarrow c$ $\bar{b} \Rightarrow \bar{c}$	$c \Rightarrow b$ $\bar{c} \Rightarrow \bar{b}$	$d \Rightarrow b$ $\bar{d} \Rightarrow \bar{b}$	$e \Rightarrow b$ $\bar{e} \Rightarrow \bar{b}$	$f \Rightarrow b$ $\bar{f} \Rightarrow \bar{b}$	$g \Rightarrow b$ $\bar{g} \Rightarrow \bar{b}$	$h \Rightarrow b$ $\bar{h} \Rightarrow \bar{b}$	$i \Rightarrow b$ $\bar{i} \Rightarrow \bar{b}$	$j \Rightarrow b$ $\bar{j} \Rightarrow \bar{b}$	$k \Rightarrow b$ $\bar{k} \Rightarrow \bar{b}$
0.81 0.86	0.86 0.86	0.90 0.90	0.69 0.69	0.90 0.90	0.59 0.59	0.68 0.68	0.63 0.63	0.63 0.63	0.60 0.60	0.86 0.86
$a \Rightarrow d$ $\bar{a} \Rightarrow \bar{d}$	$b \Rightarrow d$ $\bar{b} \Rightarrow \bar{d}$	$c \Rightarrow d$ $\bar{c} \Rightarrow \bar{d}$	$d \Rightarrow c$ $\bar{d} \Rightarrow \bar{c}$	$e \Rightarrow c$ $\bar{e} \Rightarrow \bar{c}$	$f \Rightarrow c$ $\bar{f} \Rightarrow \bar{c}$	$g \Rightarrow c$ $\bar{g} \Rightarrow \bar{c}$	$h \Rightarrow c$ $\bar{h} \Rightarrow \bar{c}$	$i \Rightarrow c$ $\bar{i} \Rightarrow \bar{c}$	$j \Rightarrow c$ $\bar{j} \Rightarrow \bar{c}$	$k \Rightarrow c$ $\bar{k} \Rightarrow \bar{c}$
0.52 0.64	0.55 0.64	0.54 0.66	0.56 0.64	0.78 0.83	0.45 0.56	0.52 0.61	0.50 0.60	0.44 0.55	0.46 0.56	0.71 0.76
$a \Rightarrow e$ $\bar{a} \Rightarrow \bar{e}$	$b \Rightarrow e$ $\bar{b} \Rightarrow \bar{e}$	$c \Rightarrow e$ $\bar{c} \Rightarrow \bar{e}$	$d \Rightarrow e$ $\bar{d} \Rightarrow \bar{e}$	$e \Rightarrow d$ $\bar{e} \Rightarrow \bar{d}$	$f \Rightarrow d$ $\bar{f} \Rightarrow \bar{d}$	$g \Rightarrow d$ $\bar{g} \Rightarrow \bar{d}$	$h \Rightarrow d$ $\bar{h} \Rightarrow \bar{d}$	$i \Rightarrow d$ $\bar{i} \Rightarrow \bar{d}$	$j \Rightarrow d$ $\bar{j} \Rightarrow \bar{d}$	$k \Rightarrow d$ $\bar{k} \Rightarrow \bar{d}$
0.83 0.84	0.86 0.80	0.79 0.82	0.54 0.62	0.52 0.64	0.55 0.67	0.75 0.83	0.53 0.65	0.46 0.59	0.52 0.65	0.50 0.62
$a \Rightarrow f$ $\bar{a} \Rightarrow \bar{f}$	$b \Rightarrow f$ $\bar{b} \Rightarrow \bar{f}$	$c \Rightarrow f$ $\bar{c} \Rightarrow \bar{f}$	$d \Rightarrow f$ $\bar{d} \Rightarrow \bar{f}$	$e \Rightarrow f$ $\bar{e} \Rightarrow \bar{f}$	$f \Rightarrow e$ $\bar{f} \Rightarrow \bar{e}$	$g \Rightarrow e$ $\bar{g} \Rightarrow \bar{e}$	$h \Rightarrow e$ $\bar{h} \Rightarrow \bar{e}$	$i \Rightarrow e$ $\bar{i} \Rightarrow \bar{e}$	$j \Rightarrow e$ $\bar{j} \Rightarrow \bar{e}$	$k \Rightarrow e$ $\bar{k} \Rightarrow \bar{e}$
0.43 0.50	0.44 0.50	0.48 0.53	0.60 0.62	0.43 0.48	0.40 0.51	0.56 0.64	0.48 0.57	0.46 0.56	0.43 0.55	0.78 0.82
$a \Rightarrow g$ $\bar{a} \Rightarrow \bar{g}$	$b \Rightarrow g$ $\bar{b} \Rightarrow \bar{g}$	$c \Rightarrow g$ $\bar{c} \Rightarrow \bar{g}$	$d \Rightarrow g$ $\bar{d} \Rightarrow \bar{g}$	$e \Rightarrow g$ $\bar{e} \Rightarrow \bar{g}$	$f \Rightarrow g$ $\bar{f} \Rightarrow \bar{g}$	$g \Rightarrow f$ $\bar{g} \Rightarrow \bar{f}$	$h \Rightarrow f$ $\bar{h} \Rightarrow \bar{f}$	$i \Rightarrow f$ $\bar{i} \Rightarrow \bar{f}$	$j \Rightarrow f$ $\bar{j} \Rightarrow \bar{f}$	$k \Rightarrow f$ $\bar{k} \Rightarrow \bar{f}$
0.50 0.59	0.54 0.61	0.52 0.61	0.76 0.80	0.56 0.64	0.53 0.64	0.56 0.61	0.67 0.70	0.58 0.61	0.64 0.67	0.42 0.48
$a \Rightarrow h$ $\bar{a} \Rightarrow \bar{h}$	$b \Rightarrow h$ $\bar{b} \Rightarrow \bar{h}$	$c \Rightarrow h$ $\bar{c} \Rightarrow \bar{h}$	$d \Rightarrow h$ $\bar{d} \Rightarrow \bar{h}$	$e \Rightarrow h$ $\bar{e} \Rightarrow \bar{h}$	$f \Rightarrow h$ $\bar{f} \Rightarrow \bar{h}$	$g \Rightarrow h$ $\bar{g} \Rightarrow \bar{h}$	$h \Rightarrow g$ $\bar{h} \Rightarrow \bar{g}$	$i \Rightarrow g$ $\bar{i} \Rightarrow \bar{g}$	$j \Rightarrow g$ $\bar{j} \Rightarrow \bar{g}$	$k \Rightarrow g$ $\bar{k} \Rightarrow \bar{g}$
0.43 0.52	0.49 0.54	0.52 0.57	0.57 0.61	0.50 0.55	0.66 0.71	0.57 0.61	0.55 0.64	0.49 0.59	0.53 0.63	0.53 0.61
$a \Rightarrow i$ $\bar{a} \Rightarrow \bar{i}$	$b \Rightarrow i$ $\bar{b} \Rightarrow \bar{i}$	$c \Rightarrow i$ $\bar{c} \Rightarrow \bar{i}$	$d \Rightarrow i$ $\bar{d} \Rightarrow \bar{i}$	$e \Rightarrow i$ $\bar{e} \Rightarrow \bar{i}$	$f \Rightarrow i$ $\bar{f} \Rightarrow \bar{i}$	$g \Rightarrow i$ $\bar{g} \Rightarrow \bar{i}$	$h \Rightarrow i$ $\bar{h} \Rightarrow \bar{i}$	$i \Rightarrow h$ $\bar{i} \Rightarrow \bar{h}$	$j \Rightarrow h$ $\bar{j} \Rightarrow \bar{h}$	$k \Rightarrow h$ $\bar{k} \Rightarrow \bar{h}$
0.44 0.54	0.48 0.56	0.43 0.54	0.49 0.57	0.46 0.55	0.56 0.64	0.50 0.58	0.53 0.64	0.57 0.62	0.71 0.76	0.49 0.55
$a \Rightarrow j$ $\bar{a} \Rightarrow \bar{j}$	$b \Rightarrow j$ $\bar{b} \Rightarrow \bar{j}$	$c \Rightarrow j$ $\bar{c} \Rightarrow \bar{j}$	$d \Rightarrow j$ $\bar{d} \Rightarrow \bar{j}$	$e \Rightarrow j$ $\bar{e} \Rightarrow \bar{j}$	$f \Rightarrow j$ $\bar{f} \Rightarrow \bar{j}$	$g \Rightarrow j$ $\bar{g} \Rightarrow \bar{j}$	$h \Rightarrow j$ $\bar{h} \Rightarrow \bar{j}$	$i \Rightarrow j$ $\bar{i} \Rightarrow \bar{j}$	$j \Rightarrow i$ $\bar{j} \Rightarrow \bar{i}$	$k \Rightarrow j$ $\bar{k} \Rightarrow \bar{j}$
0.46 0.50	0.46 0.50	0.49 0.52	0.58 0.59	0.48 0.52	0.65 0.67	0.57 0.59	0.73 0.74	0.71 0.70	0.67 0.75	0.47 0.56
$a \Rightarrow k$ $\bar{a} \Rightarrow \bar{k}$	$b \Rightarrow k$ $\bar{b} \Rightarrow \bar{k}$	$c \Rightarrow k$ $\bar{c} \Rightarrow \bar{k}$	$d \Rightarrow k$ $\bar{d} \Rightarrow \bar{k}$	$e \Rightarrow k$ $\bar{e} \Rightarrow \bar{k}$	$f \Rightarrow k$ $\bar{f} \Rightarrow \bar{k}$	$g \Rightarrow k$ $\bar{g} \Rightarrow \bar{k}$	$h \Rightarrow k$ $\bar{h} \Rightarrow \bar{k}$	$i \Rightarrow k$ $\bar{i} \Rightarrow \bar{k}$	$j \Rightarrow k$ $\bar{j} \Rightarrow \bar{k}$	$k \Rightarrow j$ $\bar{k} \Rightarrow \bar{j}$
0.76 0.80	0.86 0.78	0.74 0.77	0.52 0.61	0.77 0.83	0.59 0.51	0.52 0.62	0.46 0.57	0.46 0.57	0.43 0.54	0.47 0.50

Assessing the Fairness of Graduation Predictions

Henry Anderson
University of Texas at Arlington
henry.anderson@uta.edu

Afshan Boodhwani
University of Texas at Arlington
afshan.boodhwani@uta.edu

Ryan S. Baker
University of Pennsylvania
rybaker@upenn.edu

ABSTRACT

Predictive and data-intensive modeling has rapidly gained prominence in many research fields over the past decade. In recent years, the fairness of analytical models has become an increasingly important question for researchers: might a model systematically underperform on certain demographics? Might its application have different impacts across different demographic groups? Recently, these questions have found their way into educational research as well, where predictive and statistical modeling is used for such purposes as predicting course completion, assisting university recruitment, and proactively offering assistance to students. If such models are potentially unfair, students may remain underserved or suffer potential harm as a result. In this paper, we demonstrate two post-hoc assessments of fairness, applied to existing models predicting student graduation. Our assessments are intended to check for, rather than proactively prevent, algorithmic bias in predictive models. The first assessment investigates whether a selected model is *equitable*: does its performance systematically differ for members of different demographic groups? The second investigates whether the decision to use one model for all individuals in a given dataset is *optimal*: does using one model for all students come at a significant loss in per-group accuracy? By assessing fairness systematically, we hope to reduce to the risk of inequities from predictive analytics in education.

1. INTRODUCTION

As educational research incorporates more automated tools for analyzing and predicting such factors as student behaviors and educational outcomes, the field must grapple with the ethical question of algorithmic fairness. Do our algorithms, powerful as they may seem, encode and reinforce existing societal inequalities? Are the predictive tools we use systematically less accurate for some demographics, and does the use of these tools risk harming members of those demographics?

In this work, we assess a set of machine learning models,

trained to predict student graduation at a public, four-year university, using several definitions of fairness. Our analysis centers on race and gender, but in principle, can be extended to include age, international status, or other categorical variables.

Our investigations are driven by two main research questions: RQ 1) Are the models *equitable*? Do they perform considerably worse on students of particular genders or ethnicities? RQ 2) Are the models *optimal*? Could we achieve substantially better accuracy by using models specific to particular genders or ethnicities?

We investigate these questions through comparisons of false positive rates, false negative rates, and overall accuracy measures, since these each represent different potential impacts of these models' use. The primary intended use of the models is to aid university advisers, and faculty, and administrators in structuring student support. This intended use of the models informs our particular definitions and questions around the fairness of these models.

2. RELATED WORK

Researchers have utilized a wide range of approaches to defining fairness. Dwork et al. [4] define fairness as *similar individuals receive similar predictions* (individual fairness). Hardt et al. [7] measures *equalized odds*, which penalizes models for performing well only on the majority of data points, and *equal opportunity*, which requires parity between groups' predictions only where the ground truth is an "advantaged" outcome (e.g., "was admitted to college," "received a promotion"). Feldman et al. [5] propose measures of *group fairness*, where the distribution of errors and impacts should not vary across groups. There are many approaches to controlling for these, and other, fairness measures, as discussed in, e.g., [2, 9].

While the exact definitions and approaches to fairness are quite varied, a common thread is that there are no unambiguously "correct" definitions of fairness, or clear ways to control for all aspects of it. The particular definitions and measures depend on the specific data, analysis, and use case.

3. FAIRNESS FOR OUR MODELS

We select our measures of fairness based on the intended applications of the models, and their potential impact. The models are intended to inform and assist advisers, mentors, faculty, and university administrators in making decisions

Henry Anderson, Afshan Boodhwani and Ryan Baker "Assessing the Fairness of Graduation Predictions" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 488 - 491

about student intervention, student support, and university policies. We focus on the advising and mentoring applications, as this is the first intended use for the models.

In this setting, we anticipate a difference in the impacts of false negative and false positive errors. False negatives (students on track to graduate who are identified as likely to not graduate) are likely to result in additional contact with advisers, and additional assistance being made available. There is little harm to the student. False positives (students not on track to graduate, but who are predicted as likely to graduate) mean a student may not receive assistance from advisers. The potential harm is much greater. We thus use both false positive and false negative rates as metrics for fairness, per RQ 1, but we are more concerned with possible disparities in the false positives.

The overall accuracy of the models may also vary across our populations. Large variations put certain populations at higher risk of being systematically underserved due to lower-quality predictions, and thus less reliable information being presented to advisers and mentors. Therefore we test if selecting one model, trained on all students, is optimal, per RQ 2, by comparing its overall accuracy to the population specific models.

4. DATA

In this paper, we assess the fairness of models predicting whether a student will graduate within six years, initially presented in a conference poster [1]. In this section, we briefly describe the data on which those models were trained; more details are available in the original presentation.

The dataset utilized was from a large, publicly-funded, R1 research university in the southern United States. It contains data on 14,706 first time in college (FTIC) undergraduate students, all of whom were admitted in Fall semesters between 2006-2012 (inclusive), and were enrolled full-time. The data contains one entry per student, summarizing their first three enrolled semesters (Fall, Spring, and Summer). The final feature set covers academic performance (e.g. GPA, credit hours completed), financial information (e.g. scholarships, unmet need), pre-admission information (e.g. SAT/ACT scores), and extra-curricular activities (e.g. Greek Life, athletics). A student's first year has been shown to be an important period for determining a student's likelihood of dropping out [8], which while not quite the inverse of graduation, is a closely related outcome.

Tables 1 and 2 shows basic descriptive statistics for the dataset. Some of the populations have very small N s. We include these populations in the fairness analyses for completeness, but we caution against drawing meaningful conclusions from them. Similarly, we caution against drawing conclusions for the Multiple Ethnicities and Foreign populations, since these are "catch-all" labels, and represent extremely diverse groups of students.

5. METHODOLOGY

5.1 Model Building

We investigate the fairness of five separate models, each trained on our dataset to predict whether a student will

Table 1: Basic descriptive statistics of the dataset, by self-reported ethnicities.

Population	N	Graduation rate
American Indian	44	27.27%
Asian	2091	61.74%
African American	2092	38.48%
Foreign	296	52.03%
Hispanic/Latino	3805	43.97%
Multiple Ethnicities	499	48.30%
Hawaiian/Pacific Islander	22	54.55%
Ethnicity Not Specified	85	35.29%
White	5772	44.51%
TOTAL	14706	46.15%

Table 2: Basic descriptive statistics of the dataset, by self-reported gender.

Population	N	Graduation rate
Female	7613	50.06%
Male	7092	41.96%
Gender Unknown	1	0.00%
TOTAL	14706	46.15%

graduate within 6 years of first enrolling at the university. Early versions of these models (except Random Forest) have been presented in poster form [1]: linear kernel Support Vector Machines (SVM), Decision Trees, Random Forests, Logistic Regressions, and scikit-learn's Stochastic Gradient Descent classifier (SGD¹).

Each model was trained on 80% of the full dataset, with 20% held out for testing. The train-test split was conducted such that the proportions of students in each demographic category were as close as possible across the folds, and the within-group and overall graduation rates were as similar as possible. Model parameters were selected using 5-fold cross-validation within the training set. Ethnicity and gender features were omitted when training the models.

5.2 Assessing Equity

We measure the equity of our models via their false positive and false negative predictions on the held-out testing set. Each comparison is made using a one-versus-rest approach: *Male* versus *non-Male*, *White* versus *non-White*, etc. To compare false positive and false negative rates, we assign each student a label of 1 (indicating a false positive/false negative prediction) or 0 (true positive/true negative). The populations are compared using a χ^2 test on the resulting binary-valued vectors, with Benjamini & Hochberg's post-hoc correction [3] applied within each combination of population and fairness metric, across algorithms.

¹scikit-learn's `SGDClassifier` model uses gradient descent to construct a hyperplane classifier. We refer to this classifier, not the general numeric optimization method, in this paper. See the scikit-learn documentation for further details: <https://scikit-learn.org/stable/modules/sgd.html>

Table 3: Results of the χ^2 tests on false negatives and false positives, by demographic. Differences shown are the overall rates for students in the listed demographic, minus the overall rates for students not in the listed demographic. Benjamini & Hochberg [3] corrected p -values are in parentheses; p -values less than 0.05 are in bold. * = $N < 500$.

	Population	DT	RF	SVM	LR	SGD
False Negatives	American Indian*	0.051 (0.496)	0.036 (0.598)	0.055 (1.652)	0.055 (0.559)	0.055 (0.826)
	Asian	0.014 (0.311)	0.000 (0.997)	0.013 (0.880)	0.013 (0.239)	0.013 (0.440)
	African American	0.011 (0.729)	0.025 (0.188)	-0.001 (1.193)	0.002 (1.423)	-0.001 (0.954)
	Foreign*	0.009 (0.731)	-0.022 (2.226)	0.013 (1.458)	0.013 (0.744)	0.013 (0.972)
	Hispanic/Latino	0.020 (0.046)	0.030 (0.007)	0.010 (0.223)	0.013 (0.181)	0.011 (0.186)
	Multiple Ethnicities*	-0.001 (0.943)	-0.006 (0.992)	0.023 (1.066)	0.023 (0.555)	0.013 (0.801)
	Hawaiian/Pacific Islander*	-0.041 (1.613)	-0.055 (2.950)	-0.036 (0.831)	-0.037 (1.105)	-0.036 (0.664)
	White	-0.030 (<0.001)	-0.035 (<0.001)	-0.020 (0.006)	-0.023 (0.002)	-0.020 (0.005)
	Female	-0.005 (0.565)	0.000 (0.966)	-0.019 (0.024)	-0.017 (0.030)	-0.017 (0.025)
	Male	0.006 (0.560)	0.000 (0.974)	0.019 (0.023)	0.017 (0.029)	0.017 (0.025)
False Positives	American Indian*	0.103 (0.366)	0.122 (1.290)	0.105 (0.438)	0.108 (0.562)	0.110 (0.810)
	Asian	-0.024 (1.139)	0.013 (0.496)	-0.024 (0.581)	-0.018 (0.448)	-0.018 (0.592)
	African American	0.001 (0.956)	-0.037 (0.245)	-0.018 (0.602)	-0.021 (0.722)	-0.012 (0.653)
	Foreign*	0.060 (1.078)	0.013 (0.967)	0.046 (0.564)	0.049 (0.778)	0.001 (0.985)
	Hispanic/Latino	-0.021 (0.304)	-0.019 (0.247)	-0.019 (0.222)	-0.028 (0.346)	-0.023 (0.329)
	Multiple Ethnicities*	-0.024 (0.870)	-0.024 (1.253)	-0.032 (2.005)	-0.019 (0.764)	-0.017 (0.656)
	Hawaiian/Pacific Islander*	0.030 (0.861)	0.049 (3.794)	0.032 (1.059)	0.035 (1.392)	0.037 (2.057)
	White	0.029 (0.038)	0.032 (0.023)	0.040 (0.012)	0.043 (0.010)	0.039 (0.009)
	Female	-0.018 (0.309)	-0.008 (0.711)	-0.021 (0.648)	-0.020 (0.359)	-0.004 (0.753)
	Male	0.019 (0.300)	0.008 (0.698)	0.021 (0.627)	0.020 (0.347)	0.004 (0.741)

Table 4: Model scores, trained on all students, evaluated against the held-out testing set. Values in parentheses are the standard error, calculated as in [6].

Model	AUC
Decision Tree	0.798 (0.008)
SVM	0.805 (0.008)
Logistic Regression	0.807 (0.008)
Random Forest	0.800 (0.008)
SGD	0.814 (0.008)

5.3 Assessing Optimality

To assess the optimality of our models (RQ 2), we compare how much accuracy is gained or lost by building separate models for each population, based on AUC ROC scores. For each population in our dataset, we compare the AUC ROC scores (calculated only on the test set) of the models when trained on all students to the AUC ROC scores of the models when trained only on that population.

6. RESULTS

6.1 Model Performance

The overall metrics, evaluated against all students in the testing set, are reported in Table 4. The models achieve consistently good performance, with high AUC ROC and F1 scores. These numbers are a good baseline of performance; if the models’ performance drops or rises considerably for any demographic, that can be taken as an indication of algorithmic bias and the need for population-specific models. AUC ROC standard errors are computed according to Hanley & McNeil’s method [6].

6.2 RQ 1: Model Equity

Table 3 shows the results of our investigation into RQ 1. The majority of the comparisons do not show significance at $p = 0.05$ after correction. Notable exceptions are White students, with consistently higher false positive and lower false negative rates across all models; Hispanic/Latino students, with consistently higher false negative rates for the tree-based models; and Male students, who have consistently higher false negative rates for hyperplane-based models (SVM, SGD, and Logistic Regression).

Since the impacts of false positives are likely to be more harmful than false negatives, we find the consistently higher false positive rate for White students to be more noteworthy than the false negative results.

6.3 RQ 2: Model Optimality

Table 5 shows the AUC scores and AUC standard errors on each population for two models: one trained on the entire dataset and tested only on students in the listed demographic group (“Whole Population Model”), and one trained only on students in the listed demographic group (“Population Specific Model”). The only instance where the performance differed by more than the standard error is the Logistic Regression model for Asian students, which saw a *decrease* in performance when using the population-specific model. This indicates that the current models are optimal in terms of population-specificity: population-specific models do not gain appreciable predictive power for any population in the dataset.

Further, this indicates that the trends identified by the models generalize across populations in the dataset, and thus, the models’ performance on all students benefits from access to

Table 5: The results of comparing the AUC ROC scores for models trained on all students (“Whole Population Model,” abbreviated “WPM”) versus just one demographic (“Population Specific Model,” abbreviated “PSM”). AUCs with non-overlapping standard error (SE) intervals are in bold.

Population		DT		RF		SVM		LR		SGD	
		PSM	WPM	PSM	WPM	PSM	WPM	PSM	WPM	PSM	WPM
African American	AUC	0.788	0.796	0.799	0.804	0.818	0.830	0.805	0.830	0.799	0.830
	SE	0.024	0.024	0.023	0.023	0.022	0.022	0.023	0.022	0.023	0.022
American Indian	AUC	0.857	0.661	0.679	0.661	0.464	0.661	0.589	0.661	0.589	0.661
	SE	0.135	0.182	0.180	0.182	0.187	0.182	0.188	0.182	0.188	0.182
Asian	AUC	0.747	0.762	0.729	0.744	0.746	0.769	0.710	0.766	0.753	0.769
	SE	0.023	0.023	0.024	0.024	0.024	0.023	0.025	0.023	0.023	0.023
Female	AUC	0.795	0.800	0.803	0.798	0.814	0.815	0.812	0.816	0.798	0.811
	SE	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011
Foreign	AUC	0.770	0.712	0.763	0.796	0.677	0.729	0.658	0.729	0.620	0.781
	SE	0.060	0.066	0.061	0.057	0.068	0.064	0.070	0.064	0.072	0.059
Hispanic/Latino	AUC	0.800	0.799	0.786	0.790	0.810	0.814	0.798	0.819	0.806	0.819
	SE	0.017	0.017	0.017	0.017	0.016	0.016	0.017	0.016	0.016	0.016
Male	AUC	0.777	0.793	0.796	0.801	0.787	0.791	0.792	0.794	0.790	0.804
	SE	0.013	0.012	0.012	0.012	0.013	0.013	0.012	0.012	0.013	0.012
Multiple Ethnicities	AUC	0.816	0.818	0.812	0.826	0.809	0.807	0.807	0.797	0.723	0.807
	SE	0.043	0.043	0.043	0.042	0.044	0.044	0.044	0.045	0.051	0.044
Hawaiian/Pacific Islander	AUC	0.667	0.750	0.750	0.750	0.833	0.750	0.833	0.750	0.417	0.750
	SE	0.265	0.239	0.239	0.239	0.201	0.239	0.201	0.239	0.287	0.239
White	AUC	0.803	0.805	0.806	0.809	0.803	0.800	0.800	0.802	0.805	0.805
	SE	0.013	0.013	0.013	0.013	0.013	0.013	0.013	0.013	0.013	0.013

the full population at training time. However, since the models primarily identified GPA and student credit hours obtained as the most important predictors [1], this trend may be specific to these variables.

7. DISCUSSION AND FUTURE WORK

We have demonstrated an approach to assessing fairness that derives definitions of fairness directly from the use cases of the models in question. We find that our models are not perfectly equitable, as the term is defined in RQ 1. However, the differences are generally small (under 5%), and with the important exception of White students, are not consistent across all models. We encourage any end-users of models that display some unfair tendencies to be cautious and mindful of the potential impact. These differences are relatively small, but it has still not been established what level of unfairness should be considered acceptable in a model with real-life implications. Perfect fairness is ideal, but difficult to achieve. Our models are, though, very optimal across groups, in terms of our definition in RQ 2. No model, or population, saw a meaningful change in per-group performance when trained only on one population.

The most important avenue for future work on this subject is investigating how the implementation of models such as these (e.g. making them available to advisers) will affect student outcomes, and whether the slight unfairness observed here will translate to real-world differences. The assessment of fairness we have performed is an attempt to pre-empt such effects, but is not a substitute for directly measuring them.

8. REFERENCES

- [1] H. Anderson, A. Boodhwani, and R. S. Baker. Predicting graduation at a public R1 university. In *Proceedings of the 9th International Learning Analytics and Knowledge Conference*, 2019.
- [2] S. Barocas and A. D. Selbst. Big data’s disparate impact. *Calif. L. Rev.*, 104:671, 2016.
- [3] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.
- [4] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.
- [5] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.
- [6] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.
- [7] M. Hardt, E. Price, N. Srebro, et al. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pages 3315–3323, 2016.
- [8] V. Tinto. Research and practice of student retention: What next? *Journal of College Student Retention: Research, Theory & Practice*, 8(1):1–19, 2006.
- [9] I. Zliobaite. A survey on measuring indirect discrimination in machine learning. *arXiv preprint arXiv:1511.00148*, 2015.

Hello? Who is posting, who is answering, and who is succeeding in Massive Open Online Courses

J. Miguel L. Andres-Bray
University of Pennsylvania
3700 Walnut Street
Philadelphia, PA 19103
miggyandresbray@gmail.com

Jaclyn L. Ocumpaugh
University of Pennsylvania
3700 Walnut Street
Philadelphia, PA 19103
jlocumpaugh@gmail.com

Ryan S. Baker
University of Pennsylvania
3700 Walnut Street
Philadelphia, PA 19103
ryanshaunbaker@gmail.com

ABSTRACT

Research has shown that participating in online discussion forums is tied to improved learning outcomes, but in the case of Massive Open Online Courses (MOOCs), very few students utilize this resource. This paper presents a preliminary investigation of demographic differences in the relationship between participation (using four different measures related to students production in the forum and the responses they receive) and course completion, demonstrating the need of further research into the potential causes for these differences.

Keywords

MOOC, MORF, demographic analysis, gender, race, production rules

1. INTRODUCTION

As educators seek to better support students in Massive Open Online Courses (MOOCs), research has shown that participation in MOOC discussion forums is tied to completion rates as well as other measures of learning [4, 9, 21, 26]. However, within the same body of research, we find that the vast majority of students never post even once [20].

These low rates of participation are perhaps not surprising. In addition to factors that can make it difficult for, say, a working adult to commit to a course [13], issues surrounding the social practices of both learning and language make the sheer size of a MOOC discussion forum difficult for people to navigate [2, 19]. Some researchers, concerned about the importance of having a social presence (see [22]) in online learning, have even suggested that we try to facilitate the community structure by encouraging MOOC participants enroll with a cohort of friends (e.g., [6]).

In addition to the social factors that influence motivation, encouraging students to enroll with friends may also improve students' opportunities to learn by giving them the opportunity to speak (or write) with people who use familiar communication practices. For example, we know that cultures differ in terms of several language practices that are likely relevant to learning contexts, including appropriate manners for giving advice [29], request strategies [30], expressing disagreement [31], including

outsiders in the conversation [14] and even the interpretation of silence [10]. We also know that these differences can lead to cultural differences in broader interpretations of politeness (e.g., [7, 23]), and that sometimes these differences are even found across dialects of the same language (e.g., [18]) or across demographic groups, especially gender, within the same dialect (e.g., [17]).

Research on demographics in MOOCs has found that they can influence the level of student participation. For example, Huang et al. [11] found different rates of postings based on nationality and gender, with men posting more frequently than women and students from some countries posting at much higher rates than those from other countries. Likewise, Hodgson & Hui [12] also found differences based on nationality, with Chinese students posting at higher rates than those from other countries.

This study builds on these previous findings, exploring the role of demographic differences (namely race and gender) within the MOOC Replication Framework (MORF) [5]. Specifically, we are interested in two questions: 1) How do previous findings on forum posting behaviors and their relationship to course completion replicate across multiple MOOC sessions when broken down by reported gender and race? 2) Are these replications significantly different by race and gender?

2. METHODS

This research was conducted within the MOOC Replication Framework (MORF) [5], a platform designed to enhance large-scale replication efforts by representing previously published findings as production rules. In order to accomplish this, MORF uses a simple formalism that was previously employed in work to develop human-understandable computational theory in psychology and education [3, 16]. This approach allows findings to be represented in a fashion that human researchers and practitioners can easily understand.

All findings are converted into if-else production rules following the format, "If a student who is <attribute> does <operator>, then <outcome>." Attributes are pieces of information about a student, such as gender or race. Operators are actions a student does within the MOOC. Outcomes can represent a number of indicators of student success or failure including watching a majority of videos (e.g., [15, 24]) or publishing a scientific paper after participating in the MOOC (e.g., [26]). In the current study, we focus on the platform's most commonly-studied student outcome: whether or not the students in question completed the MOOC.

Each production rule analysis returns two counts: 1) the confidence [1], or the number of participants who fit the rule, i.e., meet both the if and the then statements, and 2) the

Juan Miguel Andres-Bray, Jaclyn Ocumpaugh and Ryan S. Baker "Hello? Who is posting, who is answering, and who is succeeding in Massive Open Online Courses" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 492 - 495

conviction [8], the production rule’s counterfactual, i.e., the number of participants who match the rule’s then statement but not the rule’s if statement. For example, in the production rule, “If a student started more threads than the session average, then they are more likely to complete the MOOC,” the two counts returned are the number of participants that started more threads than the average and completed the MOOC, and the number of participants who started threads less than the average, but still completed the MOOC. As a result, for each MOOC, a confidence and a conviction for each production rule can be generated.

A chi-square test of independence can then be calculated comparing each confidence to each conviction. The chi-square test can determine whether the two values are significantly different from each other, and in doing so, determine whether the production rule or its counterfactual significantly generalized to the data set.

In this study, we tested the replicability of four previously published findings on discussion forum posting behavior, which can be put into two categories: initiating discussion and receiving uptake of discussion. The previously published findings can be found in Table 1 as production rules. These production rules use normalized measures of students forum activity (i.e., higher or lower than average) to investigate who is responsible for starting the most threads, posts more often overall, who gets more respondents on their own threads, and who posts more responses to others’ threads.

Table 1. Categories of production rules used in the study

Condition	Source
Initiating Discussion:	
• If the student started more threads than the session average (Threads)	[4]
• If the student posts more frequently than the session average (Posts)	[9, 28]
Receiving Uptake of Discussion:	
• If the student has more respondents on their own threads than the session average (Respondents)	[21]
• If the student has more responses than the session average (Responses)	[27]

Note. The THEN clause of each production rule states, “then they are more likely to complete the course and earn a certificate.”

These findings were turned into production rules, and executed in MORF against its data store of 100 MOOC sessions, comprised of 45 different courses offered by a University on Coursera. In integrating across MOOCs, we choose the conservative and straightforward Stouffer’s [25] Z-score method to combine the results per finding across the multiple MOOC data sets, which we used to obtain a single statistical significance result across all MOOCs per reported gender and race.

Stouffer’s Z-score method was used to compare resulting Z-scores across both genders and across the four races included in the study: White or Caucasian, Black or African American, Asian, and LatinX. In this study, we included only students who

had reported their gender or their race in Coursra’s optional demographics survey. Furthermore, only students from the U.S. were included in order to avoid complications of racial/ethnic categories that do not always translate in a straightforward manner across national boundaries. Table 2 summarizes the number of students in each category.

Table 2. Summary of research subjects by race and gender categories; all included reported being U.S. students

Demographic	N
Female	10,813
Male	14,394
White or Caucasian (W)	12,802
Black or African American (B)	1,089
Asian (A)	5,299
LatinX (L)	3,960

3. RESULTS

3.1 Gender Differences

The relationship between gender and the four discussion forum conditions included in this study can be found in Table 3. Each row represents the result of testing that condition, or MORF production rule, across the full set of MOOCs, reported by gender. For example, “Posts” refers to the production rule, “If the student’s total number of posts was higher than the session average, then they are more likely to complete the course and earn a completion certificate.” The *Female* and *Male* columns list the cumulative Z-scores per reported gender, and can be interpreted as how well each rule replicated across MORF’s data. The *Diff* column lists the resulting Z-score when comparing whether the *Female* and *Male* scores are significantly different.

Table 3. Results of production rule analysis and gender differences in posting behavior (* $p < 0.001$)

Feature	Female	Male	Diff
Threads	16.36*	26.57*	7.22*
Posts	19.66*	29.89*	7.23*
Respondents	15.19*	21.69*	4.60*
Responses	14.60*	19.29*	3.32*

As seen in Table 3, all four production rules replicated significantly across all MOOC sessions in both genders. That is, thread, posts, respondents, and responses conditions all show positive relationships with MOOC completion. Moreover, as the *Diff* column shows, the positive relationship between forum participation and course completion is significantly more likely to replicate on men than women across all four measures.

3.2 Racial Differences

The relationship between race and the four discussion forum conditions included in this study can be found in Tables 4 and 5. In Table 4, each row represents the result of testing each production rule across the full set of MOOCs, broken down by reported race. As in Table 3, the *Feature* column lists the

important key words of the rule's IF statement, and the other columns list the cumulative Z-scores per reported race. Like in Table 3, these values can be interpreted as significance values of how well each rule replicated across MORF's datasets. Table 4 shows that all four production rules replicated significantly across all MOOC sessions in all reported races included in this study.

Table 4. Results of production rule analysis broken down by reported race (* $p < 0.01$, ** $p < 0.001$)

Feature	White	Black	Asian	LatinX
Threads	19.57**	4.58**	8.67**	8.64**
Posts	22.78**	3.59**	9.42**	8.58**
Respondents	14.73**	3.12*	8.18**	7.11**
Responses	14.31**	4.27**	8.11**	6.31**

Table 5 reports the differences by race in the resulting Z-score. Here, it is important to note that the racial category listed first is the category with the higher Z-score value. As the table shows, nearly all of the comparisons were statistically significant, with notable exceptions found primarily in the comparison between Asian and LatinX students (the final column), although the difference between LatinX students and Black/African American students also had one condition (responses) that was not significant.

Table 5. Racial differences in posting behavior; W = White or Caucasian, B = Black or African American, A = Asian, L = LatinX (* $p < 0.01$, ** $p < 0.001$)

Feature	W>B	W>A	W>L	L>B	A>B	A>L
Threads	10.6**	7.7**	7.7**	2.9*	2.9*	0.0
Posts	13.6**	9.5**	10.**	3.5**	4.1**	0.6
Respondents	8.2**	4.6**	5.4**	2.8*	3.6**	0.8
Responses	7.1**	4.4**	5.7**	1.4	2.7*	1.3

4. DISCUSSION & CONCLUSIONS

In order to interpret these results, it is useful to consider the semantics of these conditions, two of which (posts and threads started) look at the initiation of discussion, measuring a student's production of discussion posts and two of which (respondents and responses) measure the uptake a student receives from those in his or her cohort. Our results show all four production rules significantly replicated across all MOOC datasets, and these findings are consistent no matter the demographic category considered.

At the same time, these results show that there are demographic differences both in who chooses to post and in whose posts receive the most attention. Namely, men are more likely than women to complete the course and earn a certificate based on initiating discussion and receiving uptake, results which may, in fact, influence one another.

We found the same kind of discrepancies when we looked at racial categories. Specifically, White/Caucasian students were more likely to succeed in MOOCs based on posting and receiving uptake than any other racial category. However, while Asian or LatinX students were more likely to succeed than

Black/African American students based on forum behavior, the magnitudes of these differences were smaller than those between White/Caucasian students and other categories. Finally, Black/African American students were the least likely to succeed across all four measures.

More research is needed to better understand what might be causing these communication differences, but this preliminary study highlights the importance of examining these differences, as it shows they are important factors for predicting student success in MOOC platforms. If demographic differences exist at this level of analyses, it is likely that sociocultural strategies related to politeness or other communication practices are at play.

In future work, we hope to offer more sophisticated analyses of these issues, explicating the specific practices that may be contributing to these disparities. However, in the meantime, we hope this preliminary work serves to demonstrate a potential importance of these considerations when trying to support online learning.

5. REFERENCES

- [1] Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining Associations between Sets of Items in Massive Databases. In *Proceedings of the ACM-SIGMOD Int'l Conference on Management of Data* (pp. 207-216).
- [2] Anderson, T., & Kanuka, H. (1998). Online social interchange, discord, and knowledge construction.
- [3] Anderson, J. R., Matessa, M., & Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, 12(4), 439-462.
- [4] Andres, J.M.L., Baker, R.S., Gašević, D., Siemens, G., Crossley, S.A. and Joksimović, S. (2018), March. Studying MOOC completion at scale using the MOOC replication framework. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge* (pp. 71-78). ACM.
- [5] Andres, J. M. L., Baker, R. S., Siemens, G., Gašević, D., & Spann, C. A. (2016). Replicating 21 findings on student success in online learning. *Technology, Instruction, Cognition, and Learning*, 313-333.
- [6] Bayeck, R. (2016). Exploratory study of MOOC learners' demographics and motivation: The case of students involved in groups. *Open Praxis*, 8(3), 223-233.
- [7] Beebe, L. M., & Takahashi, T. (1989). Sociolinguistic variation in face-threatening speech acts. In *The dynamic interlanguage* (pp. 199-218). Springer, Boston, MA.
- [8] Brin, S., Motwani, R., Ullman, J. D., & Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. In *ACM SIGMOD Record* (Vol. 26, No. 2, pp. 255-264). ACM.
- [9] Crossley, S., McNamara, D. S., Baker, R., Wang, Y., Paquette, L., Barnes, T., & Bergner, Y. (2015). Language to Completion: Success in an Educational Data Mining Massive Open Online Class. *International Educational Data Mining Society*.

- [10] Duff, P. A. (2007). Problematising academic discourse socialisation. *Learning discourses and the discourses of learning*, 1(1), 1-18.
- [11] Huang, J., Dasgupta, A., Ghosh, A., Manning, J., & Sanders, M. (2014). Superposter behavior in MOOC forums. In *Proceedings of the first ACM conference on Learning@ scale conference* (pp. 117-126). ACM.
- [12] Hodgson, P., & Hui, B. (2017). The Social Presence of Active Learners across Five Countries through MOOCs: From Single to Super Postings.
- [13] Kahan, T., Soffer, T., & Nachmias, R. (2017). Types of Participant Behavior in a Massive Open Online Course. *The International Review of Research in Open and Distributed Learning*, 18(6).
- [14] Kitao, K. (1990). A Study of Japanese and American Perceptions of Politeness in Requests. *Doshisha Studies in English*, 50, 178-210.
- [15] Kim, J., Guo, P. J., Seaton, D. T., Mitros, P., Gajos, K. Z., & Miller, R. C. (2014). Understanding in-video dropouts and interaction peaks in online lecture videos. In *Proceedings of the first ACM conference on Learning@ scale conference* (pp. 31-40). ACM.
- [16] Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial intelligence*, 33(1), 1-64.
- [17] Mills, S. (2016). *Gender and Performance Anxiety. Speaking Out: The Female Voice in Public Contexts*, 61.
- [18] Murphy, M. L., & De Felice, R. (2017). Routine politeness in American and British English requests: use and non-use of please. *Journal of Politeness Research*.
- [19] Pfister, H. R., & Oehl, M. (2009). The impact of goal focus, task type and group size on synchronous net-based collaborative learning discourses. *Journal of Computer Assisted Learning*, 25(2), 161-176.
- [20] Poquet, O., & Dawson, S. 2016. Untangling MOOC Learner Networks. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*, New York, NY, USA, pp. 208–212
- [21] Ramesh, A., Goldwasser, D., Huang, B., Daumé III, H. & Getoor, L. (2013, December). Modeling learner engagement in MOOCs using probabilistic soft logic. In *NIPS Workshop on Data Driven Education* (Vol. 2, pp. 1-7).
- [22] Shelton, B. E., Hung, J. L., & Lowenthal, P. R. (2017). Predicting student success by modeling student interaction in asynchronous online courses. *Distance Education*, 38(1), 59-69.
- [23] Sifianou, M., & Blitvich, G. C. (2017). (Im) politeness and Cultural Variation. In *The Palgrave handbook of linguistic (im) politeness* (pp. 571-599). Palgrave Macmillan, London.
- [24] Sinha, T., Jermann, P., Li, N., & Dillenbourg, P. (2014). Your click decides your fate: Inferring information processing and attrition behavior from MOOC video clickstream interactions. *arXiv preprint arXiv:1407.7131*.
- [25] Stouffer, S.A., Suchman, E.A., DeVinney, L.C., Star, S.A. & Williams, R.M. Jr. (1949). *The American Soldier, Vol. 1: Adjustment during Army Life*. Princeton University Press, Princeton.
- [26] Wang, Y. Baker, R. (2015) Content or Platform: Why do students complete MOOCs? *MERLOT Journal of Online Learning and Teaching*, 11 (1), 17-30.
- [27] Yang, D., Sinha, T., Adamson, D., & Rose, C. P. (2013). Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Proceedings of the 2013 NIPS Data-driven education Workshop* (Vol. 11, p. 14)
- [28] Yang, D., Wen, M., Howley, I., Kraut, R., & Rose, C. (2015). Exploring the effect of confusion in discussion forums of massive open online courses. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale* (pp. 121-130). ACM.
- [29] Chun, M. Y. (2005). Culture-specific Concept of Politeness in Offering Advice. *proceedings of the 10th Pan-Pasific Association of Applied Linguistics, Japan. PAAL Journal*, 287-296.
- [30] Veysi, E., & Abbaszadeh, F. (2016). A Comparative Study on Intonational Elements and Commissive Illocutionary Force Interface with Respect to Contextual Aspects and Cultural Background. *Modern Journal of Language Teaching Methods*, 6(4), 387.
- [31] Zhu, W., & Boxer, D. (2012). Disagreement and sociolinguistic variables. *Pragmatic variation in first and second language contexts: Methodological issues*, 31, 113.

Augmenting Transcripts with Natural Language Processing and Multimodal Data

Tyler Angert

Harvard Graduate School of Education

tangert@gse.harvard.edu

Bertrand Schneider

Harvard Graduate School of Education

bertrand_schneider@gse.harvard.edu

ABSTRACT

In this paper we explore preliminary applications of augmenting transcripts with multimodal data. In a previous study, pairs of participants (dyads) learned how to program a robot to navigate a maze using a block-based programming language. As the dyads completed the task, their transcripts and various multimodal data were captured, creating a synced dataset of speech, 3D motion capture points, electrodermal activity, heart rate, and eye tracking. In this paper, we describe a simple visualization method to more easily analyze conversation during collaboration: a “Convergence chart” which visualizes changes in biometrics between speakers throughout a conversation. These visualizations allow researchers to see high level trends in transcripts by using a combination of Natural Language Processing (NLP) and physiological data to generate new insights on how collaboration changes over time. We conclude with how to use these visualizations to create new metrics to understand group dynamics.

Keywords

Data Visualization, Computer Supported Collaborative Learning, Natural Language Processing, Biometrics

1. INTRODUCTION

Transcripts are one of the most common forms of recording speech and are a cornerstone of qualitative data analysis. With the increase of labeled text data sets and machine learning techniques, various data visualization applications have made text and conversations easier to analyze. However, transcripts often ignore a significant amount of context from conversation such as body language, tone, shift in conversation topic, eye movements, physiological changes and physical actions. Consequently, most transcript visualization methods fail to provide adequate context for how conversation changes over time. More importantly, because it is becoming easier to collect large multimodal datasets, researchers have an open space to improve how we use multimodal data to enrich textual analysis.

Tyler Angert and Bertrand Schneider "Augmenting Transcripts with Natural Language Processing and Multimodal Data" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 496 - 499

2. LITERATURE REVIEW

2.1 General theoretical Framework

The theoretical framework for this paper comes from theories of collaborative learning, which has been one of the main topics of interest in the Learning Sciences over the last decades. In this paper, we focus on Roschelle’s [4] framework of convergent conceptual change. In this framework, collaboration is seen as the process of constructing shared meanings for conversations, concepts and experiences. This process has been extensively studied from a psycho-linguistic perspective and is referred to as grounding [5]. Building a common ground ensures that collaborators are on the same page and share a common definition of the terms used. From this perspective, grounding allows group member to anticipate and prevent misunderstanding. We use this framework as a foundation for the visualizations presented below: our hypothesis is that convergent conceptual change is punctuated by increased levels of synchronization between participants across various metrics (e.g., not just in terms of the words used, but also in terms of additional text features and multimodal indicators). The visualizations below provides a first step toward exploring how collaboration unfolds over time through a variety of metrics.

2.2 Augmenting Transcripts with Natural Language Processing and Multimodal Datasets

This paper relies heavily on previous work done in Natural Language Processing (NLP), conversation analysis, social network analysis, data visualization and Multimodal Learning Analytics (MMLA) [6]. We discuss the shortcomings and insights from related literature and how using multimodal data in transcript visualizations can be useful to generate new insights about collaborative learning. In particular, previous studies have used focused on for evaluating social dynamics in conversations with basic NLP data. For example, Viegas and Donath [1] found that using simple abstract shapes like circles and rectangles could easily reveal underlying patterns in online conversations like lurking, inactive users, and people who dominated conversations, all of which contributed to creating “a snapshot of an entire conversation in one image.”

However, relatively few studies combine NLP and biometric data directly into the visualization of dialogue. We propose that augmenting traditional data visualization techniques with multimodal data can make conversation analysis more efficient and form a foundation for future multimodal data visualization for use in other fields. In the next section, we describe the visualization methods we designed and provide specific examples

from the dataset that help us identify metrics that differentiate between productive from less productive groups.

3. METHODS

3.1 Description of the dataset

Random pairs of participants with no self-reported prior programming or robotics knowledge were tasked with programming a robot to solve a series of increasingly complex mazes in 30 minutes. Mobile eye-trackers recorded participant gaze data, bracelets captured electrodermal activity (EDA), and a motion sensor collected movement and position data. Four physiological synchrony measures were calculated for movement and EDA data: Signal Matching (SM), Instantaneous Derivative Matching (IDM), Directional Agreement (DA) and Pearson's Correlation (PC). For a description of these metrics, see [7]. All speech and verbal interactions were transcribed and labeled with timestamps. Through a series of pre processing techniques, the transcripts, gaze data, and EDA data were all aligned along each transcript timestamp. In order to compensate for the higher frequency of data from the sensors relative to the transcripts, we took the average of all of the sensor values from after each transcript line began to an ending point based on an estimated words per minute (WPM) measurement based on the average amount of time each participant spoke.

We collected three dependent measures: learning gains, collaboration quality and task performance. Learning gains were measured through a pre and post-test assessing the participants' understanding of computational concepts used in the activity. Collaboration was assessed on nine scales: sustaining mutual understanding, dialogue management, information pooling, reaching consensus, task division, time management, technical coordination, reciprocal interaction, and individual task orientation. Task performance was assessed through the quality of the final code produced by the dyad. For more information about these metrics, see [7].

For each of the transcripts, a series of basic linguistic measures were taken to serve as a baseline to compare transcripts. We hypothesized that certain linguistic features would help guide analysis of collaboration, such as word count, for past, present, and future verb usage count, singular and plural pronoun usage count, question count, and exclamation count.

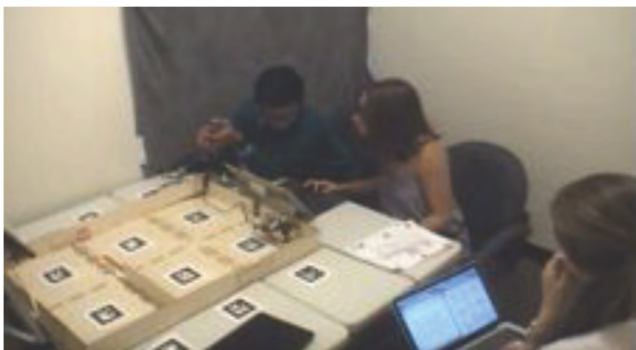


Figure 1: Example of a standard setup for two participants learning to program the robot to complete the maze tasks.

To obtain a general overview of the topics spoken for each dyad and to add more depth to the visualizations, we ran Latent Dirichlet Allocation (LDA), a topic clustering algorithm, on all of the compiled transcript lines. In order to clean the data set, we removed "stop-words" from each transcript line (e.g. "a", "the", "and"), and any speech from the facilitator of the experiment. All natural language processing was done using the Python *Gensim* and *NLTK* libraries. The following topic clusters' *top words* (TW) were calculated along with a *possible theme* (PT):

1. **PT: Analysis.** TW: **Think, need, get**, else, got, really, something, block, good, second
2. **PT: Brainstorm.** TW: **Maybe, see, try**, one, put, wait, know, want, time, could
3. **PT: Sensors / actuators.** TW: **Sensor, one, two**, motor, right, four, greater, talk, three, know
4. **PT: Adjustment.** TW: **Make, sure, know**, yes, hundred, start, sense, value, function, sorry
5. **PT: Coding.** TW: **Turn, right, go**, forward, left, want, stop, going, need, wall

3.2 Data Visualizations

To understand how we conversations change relative to biometric signals, we created a visualization to study how subjects' physiological metrics align with each other with a *Convergence chart*. In order to create the visualizations, we created a custom web-based interface that allows us to dynamically manipulate the transcript data, compare groups, and adjust the parameters of the visualizations.

The basic visualization concept behind the Convergence chart is "shifting the conversation" closer to the center as both speakers' biometrics begin to align. Below is an image that describes the concept of the visualization:

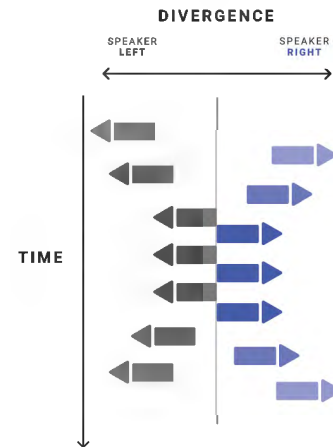


Figure 2: Schematic of the Convergence chart. Each line in the transcript is represented by a colored block.

As the participants converge towards some biometric, e.g. begin to look at the same area of a computer screen (indicating Joint Visual Attention), the transcript blocks shift closer to the center. If the participants are looking at different areas of a computer screen, the transcript blocks will diverge away from the center. We chose Joint Visual Attention (JVA) as our primary metric to analyze. JVA measures when two participants are looking at the

same area at the same time and is significantly correlated with collaboration quality [2]. We also used Signal Matching (SM) and Directional Agreement (DA) from the electrodermal data as convergence metrics [3]. SM and DA capture physiological synchrony, i.e., when two participants increase or decrease their physiological arousal at the same time. The plot moves closer to the center based on how “far away” each speaker is from a reference metric. E.g., if the metric is JVA, then each transcript block will shift towards the chart center based on how close the group’s JVA value at that timestamp is to the *average of the maximum JVA across all groups*. Global measures allow us to compare groups relative to a summarized metric. We can also choose to observe local metrics where each chart is plotted according to the five-number summary of JVA values *within* each group. The reference metric can be any metric that describes the state of the dyad, not the individual participants.



Figure 3: JVA convergence chart for Group 15 (lowest collaboration) and Group 8 (highest collaboration). Colored by topic distribution.

When we inspect closer into the JVA chart, we can glimpse into the topics and specific quotes of the conversation. Because the visualization is combined with the colored topic distributions, we gain much richer insight into *who* is speaking when JVA is highest, *what* they speak about, and the surrounding context.

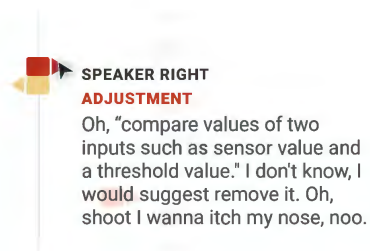


Figure 4: Focused transcript analysis from Group 15 and 8, respectively.

When we zoom out to observe patterns in larger chunks of the conversation, we can see that Group 15’s participants often take turns talking about predominantly one topic.

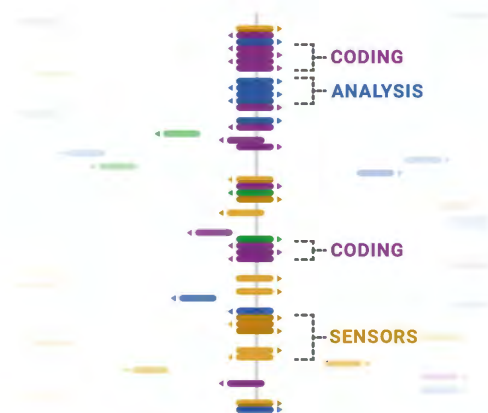


Figure 5: Conversation chunk analysis from Group 8.

Similar patterns emerge while analyzing other physiological synchrony metrics, like Directional Agreement (DA) and Signal Matching (SM) in the charts in Fig. 6.

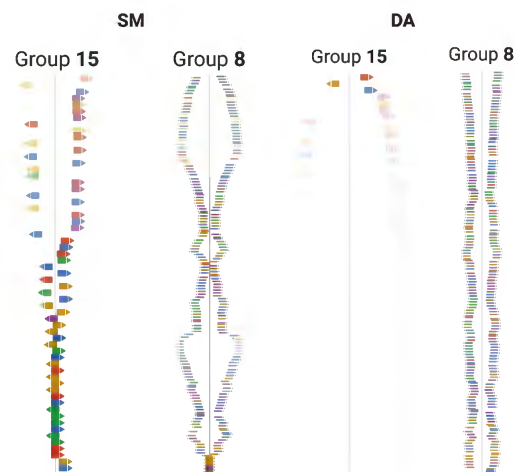


Figure 6: SM and DA convergence charts, respectively, for Group 15 (lowest collaboration) and Group 8 (highest collaboration). Colored by topic distribution.

A primary trend emerges from these charts. Even across different physiological metrics, more productive groups tend to converge towards the center of the chart more frequently than less productive groups. Researchers can consequently see conversation trends relative to attentional and physiological synchrony. These trends imply that finding the most dominant topic chunks of a conversation could correlate with or predict changes in physiological metrics.

4. DISCUSSION

The preliminary results of these visualizations illustrate the benefits of augmenting transcripts with multimodal data and topic modeling algorithms. In particular, we found that these visualizations can distinguish productive groups from less productive groups in terms of overall collaboration-related metrics on our dataset. The insight for using these graphs as a heuristic to analyze collaboration in group is simple: productive, collaborative groups' conversations will converge together more towards physiological metrics. With the Convergence chart, we see that more productive groups speak about more similar topics during periods of high JVA than less productive groups. Productive groups behaviors appear more consistent and "focused" during periods of high physiological synchrony. We believe these observed differences relate much more to broader patterns during larger chunks of conversations rather than more granular, line-by-line analysis of transcripts.

5. LIMITATIONS

These visualizations have several limitations. The current methods are only applicable to conversations with exactly two people, i.e. when dialogue is the primary method of communication. While it is possible to extend the metrics produced by the chart into more than two dimensions, new visualizations would have to be created in order to visualize actions between multiple parties in the same conversation. Further, the Convergence chart condense nearly 30 minutes of dialogue and sensor information into one graphic, which makes large or dense conversations hard to view without a large display.

6. FUTURE WORK

Our methods bring up some interesting questions about conversation and dialogue that we can hopefully answer with new metrics, in particular determining how quickly conversations change, what causes change in conversation, and how conversations progress over time.

Some ideas for answering these questions involve calculating new metrics from the visualizations, including *the comparison metric*, *comparison value*, and *the percent each transcript line converges towards the center*. Using these newly constructed data sets, supervised machine learning models could be created to predict how speech and topics could change over the course of these conversations based on changes in physiological metrics. In the future, we plan on enriching our visualizations and analyses with Coh-matrix, which provide more in depth NLP analyses of text with measures such as lexical diversity and cohesion. It would also be possible to create a general framework for auto-generating Convergence charts given multimodal dataset. We will also evaluate the visualizations' effectiveness across several domains.

7. CONCLUSION

This work provides two significant contributions. First, we developed a new visualization method to analyze conversations and augment transcripts with multimodal data. We believe that researchers can take advantage of these methods to discover high-level patterns in other transcripts and textual data sets. Second, we showed that these visualizations can potentially serve as a baseline for creating new metrics to analyze conversations, dialogue, and collaboration. We hope that the work in this paper can encourage future work on combining multimodal data with textual data visualization to enrich conversation and collaboration analysis. We see learning analytics and data mining methods as a way to bridge the gap between qualitative and quantitative research. There is an opportunity to augment both strands of research through multidisciplinary work: computational methods can facilitate manual analyses of qualitative researchers, while qualitative observations help quantitative researchers provide a window into learners' interactions. In the decades to come, computational methods can play a major role in creating synergy between qualitative and quantitative research. This paper makes a step toward bringing these two strands of research together through combining data visualization, Natural Language Processing, and Multimodal Learning Analytics.

8. ACKNOWLEDGEMENTS

We would like to thank the rest of the members of the Learning, Innovation, and Technology lab for their contributions to the initial study used in the development of this visualization.

9. REFERENCES

- [1] Viégas, F.B. and Donath, J.S. (1995). Chat circles. In the SIGCHI conference on Human Factors in Computing Systems.
- [2] Schneider, B.(accepted). Unpacking Collaborative Learning Processes during Hands-on Activities using Mobile Eye-Tracking. In the 13th International Conference on Computer Supported Collaborative Learning.
- [3] Dich, Y., Reilly, J. and Schneider, B. (2018). Using Physiological Synchrony as an Indicator of Collaboration Quality, Task Performance and Learning. In International Conference on Artificial Intelligence in Education (pp. 98-110). Springer, Cham.
- [4] Roschelle, J. (1992). Learning by collaborating: Convergent conceptual change. *The journal of the learning sciences*, 2(3), 235-276.
- [5] Baker, M., Hansen, T., Joiner, R., & Traum, D. (1999). The role of grounding in collaborative learning tasks. *Collaborative learning: Cognitive and computational approaches*, 31, 63.
- [6] Blikstein, P., & Worsley, M. (2016). Multimodal Learning Analytics and Education Data Mining: using computational technologies to measure complex learning tasks. *Journal of Learning Analytics*, 3(2), 220-238.
- [7] Starr, E., Reilly, J., & Schneider, B. (2018). Toward Using Multi-Modal Learning Analytics to Support and Measure Collaboration in Co-Located Dyads. 12th International Conference of the Learning Sciences

Predicting Student Dropout in Higher Education Based on Previous Exam Results

Alexander Askinadze
Institute of Computer Science
Heinrich Heine University Düsseldorf
Universitätsstr. 1, 40225 Düsseldorf, Germany
Alexander.Askinadze@hhu.de

Stefan Conrad
Institute of Computer Science
Heinrich Heine University Düsseldorf
Universitätsstr. 1, 40225 Düsseldorf, Germany
Stefan.Conrad@uni-duesseldorf.de

ABSTRACT

Many universities are struggling with high dropout rates. It is important to recognize at-risk students as early as possible. In the event that only data of exam results is available, as much information as possible should be extracted from this data. Instead of just using the last state of a student's passed exams the entire student progress over several semesters could be used to build the predictive model. We propose a new time series representation of the study progress data and a new distance that we use in combination with an SVM kernel for the dropout prediction task. The proposed time series based approach achieves results similar to those of a non-time series based approach and can beat it starting from the fourth semester.

Keywords

student dropout prediction, time series distance

1. INTRODUCTION

Many universities are struggling with high dropout rates. According to the National Center for Educational Statistics, only about 60% of students who started their undergraduate studies in 2010 with a standard 4-year study program received a bachelor's degree from the same university within 6 years. In addition, around 80% of the students who started their studies in 2015 remained in the following year [5]. Especially in MINT study subjects, the dropout rate at the beginning of studies is particularly high. In [3], the authors report that the dropout rate of freshmen in the Electrical Engineering course is about 40%.

We have a data set of 857 computer science students from a German university who have been enrolled for at least one exam and who either have completed their studies with a bachelor's degree (421 students) or left their studies (at the university where they started) without graduation (436 students). Students who finished their studies in computer science without a bachelor's degree may also be subject chang-

ers, who have changed their main subject, or have continued their studies at another university or who have finally dropped out of their studies. We consider these 436 students as dropouts. When we speak of dropout prediction, we mean the prediction whether the students who have enrolled in computer science, for whatever reason, do not successfully complete the study at the same university. In Fig. 1 we show the distribution of the 436 dropouts in the first 10 semesters. Most dropouts are in the second semester (about 20%), so that by the end of the 2 semesters (i.e. the first year) about 25% of all dropouts have abandoned their studies. By the end of the 4th semester 50% of the dropouts have abandoned their studies and 80% by the end of the 10th.

Since most dropout students abandon their studies relatively early, it is important to recognize them as early as possible in order to start possible interventions. In [3] it is reported that human monitoring is used to solve this problem. A large number of students require considerable manual effort so automatic dropout prediction could facilitate the work.

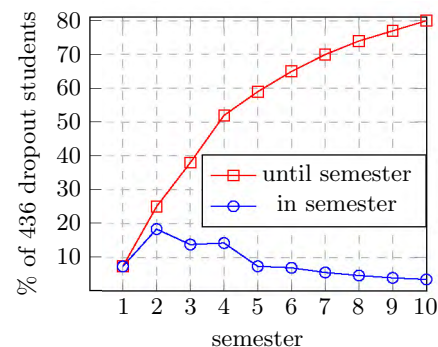


Figure 1: distribution of dropouts

For reasons of data protection (GDPR) or other reasons, universities often have only limited amounts of data about their students. Since universities are legally required to store data like results of the exams, this data could be used to create dropout prediction models. If only data of exam results is available, as much information as possible should be extracted from this data. Instead of just using the last state of a student's passed exams the entire student progress over several semesters could be used to build the predictive models. The Sankey diagram in Fig. 2 visualizes the study progress of students that dropped out until the end of the 4th semester as well as graduates. Each node in a Sankey

Alexander Askinadze and Stefan Conrad "Predicting Student Dropout in Higher Education Based on Previous Exam Results" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 500 - 503

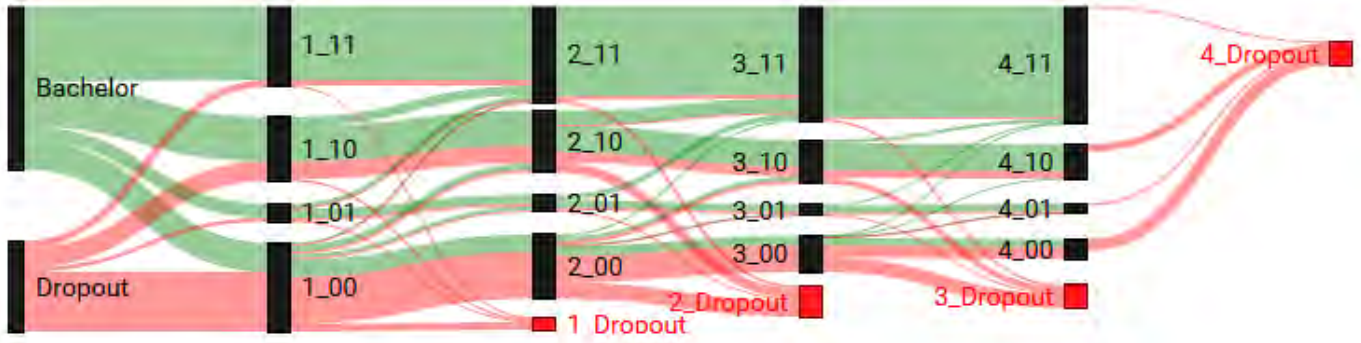


Figure 2: Sankey diagram showing study progress of graduates and students who dropped out until the fourth semester. Legend: (11) Programming + Calculus I passed (10) Programming passed (01) Calculus I passed (00) Nothing passed

diagram is a combination of exams in a specific semester. For space reasons, the data in this illustration is only coded based on the two subjects Programming and Calculus I. In the caption of the visualization is a legend, which assigns the binary numbers to their corresponding combination of passing (1) and failing (0) exams. For example, the notation 110 represents those students that only passed Programming in the first semester. The visualization clearly shows that, based on the passed exam combinations, dropouts and graduates can have very different paths. This raises the core question of this study: How to create dropout prediction models based on previous exams, if not only the last state of passed exams is used, but also the students' entire study progress should be included? To do this, we will use the dynamic time warping distance and another proposed distance and show how the time series distances can be used in the kernel of an SVM for dropout prediction.

2. RELATED WORK

Many studies have been published on student dropout prediction using classifiers such as Support Vector Machine (SVM), Random Forest (RF), CART, C4.5, k-NN and Naive Bayes [1, 3, 4]. Since the data used varies widely and the data sets are often not published due to privacy concerns, it is often difficult to compare the results of different studies. A recent study [4] compares different strategies to numerically represent the student data. The authors divide the approaches of the representations for the student dropout prediction task into 3 areas: 1) Global Feature-Based (GFB) (features that can be heard from the student of the degree to which the student belongs); 2) Local Feature-Based (LFB) (Opting to use features that apply only to a particular degree results); and 3) Time Series (TS) (using the time information of the student's data to enrich the representation). The authors conclude that TS based approaches are less well suited. In this paper, we show how to improve a TS Distance-Based (TS-DB) approach presented in [1], so that this approach delivers better results.

3. METHOD

In this chapter, it is shown how a student study progress is transformed into a multivariate time series and how time series distances can be used to create SVM Kernel in order to create competitive dropout prediction models.

Let S be the set of all students with at least one exam at-

tempt. Let $C = \{c_1, \dots, c_n\}, n \geq 1$ be the set of exams that we want to use to create our student representation. We then define a mapping function $\psi_t^C : S \rightarrow \{0, 1\}^n$ that transforms the data of a student $s \in S$ to an n -dimensional binary vector and differ two notations $\psi_{t=k}^C$ and $\psi_{t \leq k}^C$.

The difference between $\psi_{t=k}^C$ and $\psi_{t \leq k}^C$ is that $\psi_{t=k}^C$ only shows the information describing which exams were passed in semester k and $\psi_{t \leq k}^C$ shows the information describing which exams were passed between first and k -th semester. Each value $\psi_t^C(s)[i] \in \{0, 1\}$ is representing the information whether an exam was passed or not as visualized in Fig. 3.

$$\psi_{t=k}^C(s) = \left[\underbrace{0 \text{ or } 1}_{c_1 \text{ passed in sem. } k} \quad \dots \quad \underbrace{0 \text{ or } 1}_{c_n \text{ passed in sem. } k} \right]$$

$$\psi_{t \leq k}^C(s) = \left[\underbrace{0 \text{ or } 1}_{c_1 \text{ passed until sem. } k} \quad \dots \quad \underbrace{0 \text{ or } 1}_{c_n \text{ passed until sem. } k} \right]$$

Figure 3: Visualization of a coded semester vector

Example: Having a set C of the three exams with $c_1 = \text{Calculus}$, $c_2 = \text{Linear Algebra}$ and $c_3 = \text{Programming}$ and a student s who passed Calculus in the first semester, Programming in the second and Linear Algebra in the third semester, the mappings of the two versions of ψ_t^C would be as visualized in Fig. 4

$$\begin{aligned} \psi_{t=1}^C(s) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} & \psi_{t \leq 1}^C(s) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \\ \psi_{t=2}^C(s) &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} & \psi_{t \leq 2}^C(s) &= \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \\ \psi_{t=3}^C(s) &= \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} & \psi_{t \leq 3}^C(s) &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \end{aligned}$$

$\underbrace{\quad\quad\quad}_{c_1 \quad c_2 \quad c_3}$ $\underbrace{\quad\quad\quad}_{c_1 \quad c_2 \quad c_3}$

Figure 4: Exemplary visualizations of $\psi_{t=k}^C$ and $\psi_{t \leq k}^C$

In the following, we will only use the alternative $\psi_{t \leq k}^C$ because it works better than $\psi_{t=k}^C$. $\psi_{t=k}^C$ was only introduced because it corresponds to the creation of the time series in [1] and so we can specify the exact difference in the modelling of the time series of the two approaches.

There are different ways to represent a student based on his previous examinations. A typical option would be to represent the student using the last state of his or her exam results $\psi_{t \leq k}^C$. Based on this, the distance between two students s_1 and s_2 could be calculated with $d(\psi_{t \leq 3}^C(s_1), \psi_{t \leq 3}^C(s_2))$. In this case, however, the information would disappear, as how the study progress of the students looked until then. As you can see in Fig. 2, student's study progress can be very different. Assuming that not only the final state of the exam results, but the complete study progress might have an impact on whether or not a student drops out in the future, we now define the multivariate time series for students:

DEFINITION 1. Let $C = \{c_1, \dots, c_n\}$ be a set of selected exams. Let $t_{ij} \in \{0, 1\}$ be the result of exam c_i until end of semester j of a student s . A student time series $T_k^C(s)$ representing the study progress of k semesters is defined by:

$$T_k^C(s) = [\psi_{t \leq 1}^C(s), \dots, \psi_{t \leq k}^C(s)] = \left[\begin{bmatrix} t_{11} \\ \vdots \\ t_{n1} \end{bmatrix}, \dots, \begin{bmatrix} t_{1k} \\ \vdots \\ t_{nk} \end{bmatrix} \right]$$

Thus, we can present a student as a temporal sequence of his completed semesters, each semester carrying different kinds of information. Let $T_k^C(s_1)$ and $T_k^C(s_2)$ be two multivariate sequences representing the current study progress of the two students s_1 and s_2 . To compare these two sequences, we need a similarity measure or a distance $d(T_k^C(s_1), T_k^C(s_2))$ for multivariate time series.

3.1 Time Series Distances

This subchapter shows the Dynamic Time Warping (DTW) distance and a new proposed distance for student time series.

3.1.1 Multidimensional Dynamic Time Warping

A well-researched distance for univariate time series is the DTW distance. In [6] the *multidimensional* DTW distance has been proposed and its application for the student dropout prediction in [1]. The following example shows the application of the presented model with DTW. Let s_1 and s_2 be two students, for whom the data of the first 4 semesters is available, and let $C = \{c_1, c_2, c_3, c_4\}$ be a set of 4 exams on which the two student time series (Fig. 5) are based:

$$\underbrace{\left[\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right]}_{= T_4^C(s_1)} \quad \underbrace{\left[\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right]}_{= T_4^C(s_2)}$$

Figure 5: Two exemplary student time series

The example is chosen so that both students have the same status after the fourth semester with respect to exams C: $\psi_{t \leq 4}^C(s_1) = \psi_{t \leq 4}^C(s_2)$. The distance $d(\psi_{t \leq 4}^C(s_1), \psi_{t \leq 4}^C(s_2))$ for the vectors representing the last state would in this case be 0. However, if the entire course of study is included, then the distance would have to be greater than 0. In Figure 6 we show the calculation of the DTW distance $d(T_4^C(s_1), T_4^C(s_2))$ for these two students. By including the time series, the distance between the two students is no longer 0, although at

the end they have the same results with respect to C. To

$[1 \ 1 \ 1 \ 1]$	<table><tr><td>4</td><td>3</td><td>2</td><td>0</td></tr><tr><td>4</td><td>3</td><td>2</td><td>0</td></tr><tr><td>3</td><td>2</td><td>1</td><td>1</td></tr><tr><td>2</td><td>3</td><td>2</td><td>2</td></tr></table>	4	3	2	0	4	3	2	0	3	2	1	1	2	3	2	2	$[1 \ 1 \ 1 \ 1]$	<table><tr><td>13</td><td>10</td><td>8</td><td>5</td></tr><tr><td>9</td><td>7</td><td>6</td><td>5</td></tr><tr><td>5</td><td>4</td><td>5</td><td>6</td></tr><tr><td>2</td><td>5</td><td>7</td><td>9</td></tr></table>	13	10	8	5	9	7	6	5	5	4	5	6	2	5	7	9
4	3	2	0																																
4	3	2	0																																
3	2	1	1																																
2	3	2	2																																
13	10	8	5																																
9	7	6	5																																
5	4	5	6																																
2	5	7	9																																
$[1 \ 1 \ 1 \ 1]$	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	$[1 \ 1 \ 1 \ 1]$	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1
0	0	0	1																																
0	0	1	1																																
0	1	1	1																																
0	0	0	1																																
0	0	0	1																																
0	0	1	1																																
0	1	1	1																																
0	0	0	1																																
$[1 \ 1 \ 1 \ 0]$	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	$[1 \ 1 \ 1 \ 0]$	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1
0	0	0	1																																
0	0	1	1																																
0	1	1	1																																
0	0	0	1																																
0	0	0	1																																
0	0	1	1																																
0	1	1	1																																
0	0	0	1																																
$[1 \ 1 \ 0 \ 0]$	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	$[1 \ 1 \ 0 \ 0]$	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1
0	0	0	1																																
0	0	1	1																																
0	1	1	1																																
0	0	0	1																																
0	0	0	1																																
0	0	1	1																																
0	1	1	1																																
0	0	0	1																																

(a) pairwise Manhattan distances of $T_4^C(s_1)$ and $T_4^C(s_2)$ (b) cumulative distances of $T_4^C(s_1)$ and $T_4^C(s_2)$

Figure 6: MD-DTW example for 2 students' time series

calculate the distance of the whole time series, a cumulative distance matrix (Fig. 6b) is calculated, based on the calculated distances of the pairs of the time units (Fig. 6a). The values at the index (i, j) of the cumulated matrix are calculated by a cost function γ , which is based on dynamic programming and is defined recursively in equation 1. The last calculated value $\gamma(4, 4) = 5$ indicates the DTW distance.

$$\gamma(i, j) = d(a_i, b_j) + \min \left\{ \begin{array}{l} \gamma(i-1, j), \\ \gamma(i-1, j-1), \\ \gamma(i, j-1) \end{array} \right\} \quad (1)$$

3.1.2 Weighted Semester Distance

Here we present another approach to the distance calculation of two time series with exam results. Unlike the DTW approach, only the results of the same semester are compared here. In addition, the influence of individual semesters is weighted differently. For two time series $T_k^C(s_1)$ and $T_k^C(s_2)$ we define the d_{WSD} distance:

$$d_{WSD}(T_k^C(s_1), T_k^C(s_2)) = \sum_{i=1}^k w_i d(\psi_{t \leq i}^C(s_1), \psi_{t \leq i}^C(s_2))$$

For k semesters, the weights w_i per semester are chosen as $w_i = \frac{i^2}{\sum_{j=1}^k j^2} = \frac{i^2}{\frac{1}{6}k(k+1)(2k+1)}$ so that $\sum_{i=1}^k w_i = 1$ and the later semesters are weighted more heavily than the first.

3.2 Time Series Kernel

A popular method for binary classification problems is the *support vector machine* (SVM). The decision rule formulation of the SVM (with L support vectors) using the kernel trick for an input vector x^* that needs to be classified is given by $f(x^*) = \text{signum} \left(\left[\sum_{i=1}^L \alpha_i y_i K(x^*, x_i) \right] - b \right)$. A well-known and often used kernel is the RBF-kernel and is defined by: $K_{RBF}(x, y) = \exp(-\gamma \|x - y\|^2)$. An adaptation of the RBF kernel to the Gaussian DTW (GDTW) [2] kernel based on the d_{DTW} distance is specified by:

$$K_{GDTW}(x, y) = \exp(-\gamma d_{DTW}(x, y))$$

Using the d_{WSD} distance we define the corresponding adaptation of the RBF kernel by:

$$K_{WSD}(x, y) = \exp(-\gamma d_{WSD}(x, y))$$

The parameter γ must be learned with the training data.

Model	2nd			3rd			4th			5th		
	Re	Pr	F_1	Re	Pr	F_1	Re	Pr	F_1	Re	Pr	F_1
TS-DB: SVM + GDTW Kernel	0.77	0.74	0.75	0.7	0.81	0.75	0.59	0.84	0.75	0.53	0.88	0.66
TS-DB: SVM + WSD Kernel	0.74	0.75	0.75	0.74	0.83	0.78	0.75	0.81	0.78	0.74	0.82	0.78
LFB: SVM + RBF Kernel	0.73	0.75	0.74	0.73	0.84	0.78	0.72	0.81	0.77	0.72	0.83	0.77
LFB: RF	0.77	0.75	0.76	0.71	0.81	0.75	0.68	0.75	0.71	0.67	0.78	0.73

Table 1: Classification results (Recall, Precision, F_1) for student data including exam results up to the end of k -th semester

4. EVALUATION

The evaluation is done on a data set with 857 students. For each student, the following information about his or her achievements is available: idCourse, exam status (passed or failed) and the semester of the exam attempt. We evaluate how well the prediction works using the data available until the end of the 2nd, 3rd, 4th or 5th semester. The 1st is not considered since no advantages of time series can be used. For example, if the prediction for data including exam results until the end of the 4th semester is evaluated, all dropouts and graduates who have studied at least 4 semesters will be considered. It follows that with increasing semesters, there are fewer and fewer students that can be used for evaluation since students drop out or graduate before and cannot contribute to the training and evaluation.

The evaluation is performed 3 times with 10-fold stratified cross-validation. We use Recall (Re) and Precision (Pr) as evaluation measures. If as many dropout students as possible should be found, Recall should be as large as possible. If it is not very important that all are found, but the results should be really endangered students, then Precision should be as high as possible. In addition, the F_1 measure is specified, taking into account both Precision and Recall.

The length of the resulting numerical representation (multivariate time series vectors or simple vectors) depends strongly on the number of courses used since the vectors are created based on selected courses. In the data set, there are more than 100 courses. As you can see in Fig. 2, just a few exams are enough to show differences in the study progress of the dropouts and graduates. Therefore, we sort all courses according to the number of students who have enrolled in them and take $3k$ courses with the highest enrollment until k -th semester for the evaluation of students who studied at least k semesters (e.g., 6 courses with the highest enrollment until the 2nd semester for students who studied at least 2 semesters). The 3 courses are added per semester, as the curriculum includes about 3 courses per semester.

In Table 1, we compare the results of the two TS-DB approaches (numerical representation of a student is $T_k^C(s)$) based on the two SVM kernels k_{GDTW} and k_{WSD} . Additionally, two LFB approaches (numerical representation of a student is $\psi_{t \leq k}^C(s)$) with the RF and SVM (RBF kernel) classifiers are evaluated. If only the exam results until the end of the 2nd semester are used to create the prediction model, the LFB approach with the RF classifier achieves the best results in all three evaluation measures. The other approaches achieve similar or slightly worse results. In the 3rd semester, the TS-DB approach with the WSD kernel reaches the best Recall and F_1 value, whereby the LFB ap-

proach with the SVM classifier achieves a slightly better Precision value. However, from the 4th semester, the TS-DB approaches can take advantage and are better in all three measures. If the Recall or the F_1 measure is more important, then the proposed WSD kernel is more useful and if Precision is more important, the GDTW kernel is better.

5. CONCLUSION

Often only the data of the study progress per semester are available. Since these data are arranged chronologically, we have shown how the study progress data can be transformed into a multivariate time series. We presented a new kernel (WSD) and showed that the TS-DB based approaches deliver better results than LF based approaches (using only the final state of the exam results) from the 4th semester onwards. In future research, we will not only include the exam results (passed or failed) in the model but also other information such as the number of attempts or the grades.

6. REFERENCES

- [1] A. Askinadze and S. Conrad. Application of the dynamic time warping distance for the student drop-out prediction on time series data. In *Proceedings of the 10th International Conference on Educational Data Mining, EDM 2017*, pages 342–343, Wuhan, Hubei, China, June 2017.
- [2] C. Bahlmann, B. Haasdonk, and H. Burkhardt. Online handwriting recognition with support vector machines - a kernel approach. In *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition*, pages 49–54, Aug 2002.
- [3] G. Dekker, M. Pechenizkiy, and J. Vleeshouwers. Predicting students drop out: A case study. In *Proceedings of the 2nd International Conference on Educational Data Mining, EDM 2009*, pages 41–50, Cordoba, Spain, July 2009.
- [4] R. Manrique, B. P. Nunes, O. Marino, M. A. Casanova, and T. Nurmikko-Fuller. An analysis of student representation, representative features and classification algorithms to predict degree dropout. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 401–410. ACM, 2019.
- [5] J. McFarland, B. Hussar, X. Wang, J. Zhang, K. Wang, A. Rathbun, A. Barmer, E. F. Cataldi, and F. B. Mann. The condition of education 2018. nces 2018-144. *National Center for Education Statistics*, 2018.
- [6] G. A. ten Holt, M. J. Reinders, and E. Hendriks. Multi-dimensional dynamic time warping for gesture recognition. In *Thirteenth annual conference of the Advanced School for Computing and Imaging, The Netherlands, 2007*, volume 300, 2007.

The Guided Team-Partitioning Problem: Definition, Complexity, and Algorithm *

Sanaz Bahargam
Boston university
bahargam@bu.edu

Theodoros Lappas
Stevens Institute of
Technology
tlappas@stevens.edu

Evimaria Terzi
Boston University
evimaria@cs.bu.edu

ABSTRACT

A long line of literature has focused on the problem of selecting a team of individuals from a large pool of candidates, such that certain constraints are respected, and a given objective function is maximized. In this work, we extend the team formation literature by introducing the Guided Team-Partitioning (GTP) problem, which asks for the partitioning of a population into teams such that the centroid of each team is as close as possible to a given target vector. This formulation allows the team-builder to control the composition of the produced teams and has natural applications in practical settings. Algorithms for the GTP need to simultaneously consider the composition of multiple non-overlapping teams that compete for the same population of candidates. This makes the problem considerably more challenging than formulations that focus on the optimization of a single team. In fact, we prove that GTP is NP-hard to solve and even to approximate. The complexity of the problem motivates us to consider efficient algorithmic heuristics, which we evaluate via experiments on both real and synthetic datasets.

Keywords

Team Formation, Partitioning

The input of the general team formation problem consists of a pool of candidates, a set of constraints, and an objective function. The goal is then to strategically select a group of individuals from the pool, such that the group respects all the constraints and also optimizes the objective function. A long line of literature has addressed different versions of the problem that ask for the optimization of functions such as the quality of intra-team communication. Previous work has also considered a diverse array of constraints, such as those on the team's size, the team-builder's recruitment budget [7], the abilities of students [3, 4], the performance of the teams [2, 1], and the compatibility of the team members [5].

*(Does NOT produce the permission block, copyright information nor page numbering). For use with ACM_PROC_ARTICLE-SP.CLS. Supported by ACM.

Sanaz Bahargam, Theodoros Lappas and Evimaria Terzi "The Guided Team-Partitioning Problem: Definition, Complexity, and Algorithm" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 504 - 507

In this paper, we extend the team formation literature by introducing an alternative formulation of the problem that asks for the partitioning of the given pool of candidates into *multiple* teams, while allowing the team builder to control the properties of each team. We refer to this paradigm as *Guided Team-Partitioning* (GTP). For instance, consider a teacher who is trying to partition the students in her class into groups, such that the distribution of talent and experience across the groups is balanced. We illustrate an example of this scenario in Figure 1. Conceptually, our goal is to

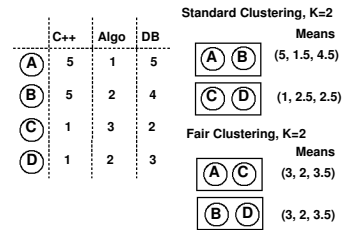


Figure 1: Standard Vs. Fair Clustering

avoid teams with an unfair advantage or disadvantage. We refer to (3, 3, 3) as the *target vector*, which is used to guide the partitioning task. We refer the interested readers to the long version of this paper for more examples [6].

1. PROBLEM DEFINITION

In this section, we begin by describing notational conventions that we will use throughout the paper; then we present the formal statement of the problems that we study. We start from a simple version of our problem with only one team. We show that even the simple version is NP-hard to solve and approximate. Then we move to partitioning problem with desired centroids. In Subsection 1.4, we describe our problem, GUIDED TEAM-PARTITIONING and we show that our problem is NP-hard to solve and approximate.

1.1 Preliminaries

We consider a pool \mathcal{R} of n individual experts. Each expert $\mathbf{r} \in \mathcal{R}$ is associated with a d -dimensional feature (skill) vector \mathbf{r}_i , such that $\mathbf{r}_i(f)$ returns the value of skill f for expert i . We also consider given a set of target vectors \mathbf{T} of k target points \mathbf{t} . The goal is to partition the pool of experts into k teams such that the cost of this partitioning is minimized. The cost for each team C_i , is the distance between the centroid of that team ($\text{mean}(C_i)$) to its target vector \mathbf{t}_i .

$$\text{Cost} = \sum_{i=1}^k \mathbf{D}(\text{mean}(C_i), \mathbf{t}_i) \quad (1)$$

We quantify the closeness between two vectors τ and t of dimension d , using the L_2^2 norm of their difference. We denote this by

$$\mathbf{D}(\tau, t) := L_2^2(\tau - t) = \sum_{i=1}^d (\tau(i) - t(i))^2$$

We define the **mean** of a set \mathcal{R} , consisting of n vectors $\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_n$ as

$$\mathbf{mean}(\mathcal{R}) = \frac{\sum_{i=1}^n \mathbf{r}_i}{n}$$

1.2 The CIS Problem

In Characteristic-Item Selection (CIS) problem, the goal is to find a subset of a universe set, such that the mean of the subset is as close as possible to a designated point. This designated point represents the whole set. This intuition is captured in a formal definition as follows.

Problem 1 (CIS). *Given a set $\mathcal{R} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_n\}$, a number ℓ , and a designated vector \mathbf{t} , find ℓ points (\mathcal{R}_ℓ) to remove from \mathcal{R} , such that **mean** of the remaining points $\mathcal{R} \setminus \mathcal{R}_\ell$ is close to target \mathbf{t} . More formally,*

$$\mathbf{D}(\mathbf{mean}(\mathcal{R} \setminus \mathcal{R}_\ell), \mathbf{t})$$

is minimized.

Lemma 1. *The CIS problem is NP-hard to solve and approximate.*

We refer the interested readers to the long version of this paper for proofs of the lemmas [6].

1.3 The CP Problem

Given a set of n workers with different skills and a set of k tasks that need to be done collaboratively with given average required skill levels for each task, the goal is to form teams of workers that their mean skill level matches each task the best. This team formation problem can be considered as one of the applications of the CP (Characteristic Partitioning) problem which is formally defined in the following problem

Problem 2 (CP). *Given a set $\mathcal{R} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_n\}$ and target vectors $\mathbf{T} = \{\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_k\}$, partition \mathcal{R} into k partitions $\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_k$ such that*

$$\sum_{i=1}^k \mathbf{D}(\mathbf{mean}(\mathcal{C}_i), \mathbf{t}_i)$$

is minimized.

Corollary 1. *The CP problem is NP-hard to solve and NP-hard to approximate.*

1.4 The GUIDED TEAM-PARTITIONING Problem

For many applications, it is not necessary to assign *all* the points in the dataset to teams. For instance, when separating a workforce into teams so that each team has a specific level of expertise in each skill, it is acceptable to exclude some of the workers. In general, having the option to exclude up to a fixed number of points adds flexibility and can only make the problem easier to solve. We capture this intuition in a formal problem definition as follows.

Problem 3 (GUIDED TEAM-PARTITIONING). *Given a set $\mathcal{R} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_n\}$, and target vectors $\mathbf{T} = \{\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_k\}$, and a budget ℓ , find ℓ points (\mathcal{R}_ℓ) to remove from the dataset and partition the rest of the points $\mathcal{R} \setminus \mathcal{R}_\ell$ into k partitions $\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_k$ such that Cost =*

$$\sum_{i=1}^k \mathbf{D}(\mathbf{mean}(\mathcal{C}_i), \mathbf{t}_i)$$

is minimized.

Corollary 2. *The GUIDED TEAM-PARTITIONING problem is NP-hard to solve and NP-hard to approximate.*

This problem is clearly NP-hard, as it contains the CP problem as a special case (for $\ell = 0$).

So far we have discussed the GUIDED TEAM-PARTITIONING problem. For a collection of workers \mathcal{R} in which each worker has a proficiency level for each skill, the most natural translation of the target vectors is the mean of the proficiencies or the required skills of a given project. For instance, consider online labor markets such as Freelancer (www.freelancer.com), Guru (www.guru.com), and oDesk (www.odesk.com) where employers hire freelancers with specific skills to work on different types of projects. The required skills listed for each project in these platforms can be used as the target vectors. In such a setting, each team should poses a specific share of expertise across all skills that makes the team capable of finishing a particular task or project.

2. ALGORITHM

In this section, we describe GUIDEDSPLIT; it finds an efficient solution for GUIDED TEAM-PARTITIONING. We start by presenting algorithms to solve CIS and CP problem and then we use these algorithms to solve GUIDED TEAM-PARTITIONING problem.

2.1 Solving The CIS Problem

Although CIS problem is NP-hard to solve and approximate, we propose heuristic algorithms that work well in practice.

The CONVEXOPT Algorithm: We can formulate CIS problem as a Mixed Integer optimization problem to find a binary vector x such that $\mathbf{mean}(\mathcal{R}x) = \mathbf{t}$ subject to $\sum_{i=1}^n x_i = n - \ell$.

Since solving mixed integer problem is NP-hard and would require complex algorithms to solve, we instead relax this problem to a convex quadratic programming by removing the binary constraints as shown in Algorithm 1. This CONVEXOPT algorithm forms a nonnegative real-valued vector x such that $\mathbf{D}(\mathbf{t}, \mathbf{mean}(\mathcal{R}x))$ (Line 1) is small. Note that the aim is to find a subset \mathbf{S} of length $n - \ell$ such that $\mathbf{mean}(\mathbf{S})$ is close to \mathbf{t} , in another word ideally, we want $n - \ell$ elements of x to be equal $\frac{1}{(n-\ell)}$, and the rest be 0 (and $\mathcal{R}x = \mathbf{t}$). This constraint is implied in Line 4 and 3. The algorithm tries to find a vector x of real values such that its elements are between 0 and 1 and the sum of elements of x is at least $n - \ell$. Then we transform this real-valued vector to a binary vector by replacing the $n - \ell$ largest elements to 1 and the rest to 0. We used CVX package in Matlab to solve this convex quadratic programming.

Algorithm 1 The CONVEXOPT Algorithm

Input: Input set \mathcal{R} , target vector \mathbf{t} and number of points to be removed ℓ
Output: A subset \mathbf{S} such that $|\mathbf{S}| = n - \ell$

- 1: minimize $\mathbf{D}(\mathcal{R} * x * \frac{1}{n-\ell}, \mathbf{t}')$
- 2: subject to
- 3: $\text{sum}(x) \geq n - \ell$
- 4: $x \geq 0$ and $x \leq 1$ and $x \in \mathbb{R}$
- 5: Set $n - \ell$ largest values of x to 1 and the rest to 0
- 6: return $\mathbf{S} = \mathcal{R}(x)$

2.2 Solving CP Problem

Our algorithm for the CP problem, which we call the MAX BENEFIT algorithm, is a polynomial time algorithm that finds a partitioning of data points \mathcal{R} , into k partitions. The pseudocode of the MAX BENEFIT algorithm is shown in Algorithm 2.

Algorithm 2 The MAX BENEFIT Algorithm

Input: Input set \mathcal{R} , target vectors \mathbf{T} and number of partitions k
Output: Partitions \mathcal{C}

- 1: $\mathcal{C} = \{\}$
- 2: **for** \mathbf{r} in \mathcal{R} **do**
- 3: $j = \text{argmax}_{i=1 \dots k} (\mathbf{D}(\mathbf{t}_i, \text{mean}(\mathcal{C}_i)) - \mathbf{D}(\mathbf{t}_i, \text{mean}(\mathcal{C}_i \cup \{\mathbf{r}\})))$
- 4: Add \mathbf{r} to partition j , \mathcal{C}_j
- 5: **while** no convergence achieved **do**
- 6: **for** \mathbf{r} in \mathcal{R} **do**
- 7: $h =$ The partition \mathbf{r} belongs to
- 8: $\text{loss} = \mathbf{D}(\mathbf{t}_h, \text{mean}(\mathcal{C}_h \setminus \{\mathbf{r}\})) - \mathbf{D}(\mathbf{t}_h, \text{mean}(\mathcal{C}_h))$
- 9: $j = \text{argmax}_{i=1 \dots k} (\text{loss} + \mathbf{D}(\mathbf{t}_i, \text{mean}(\mathcal{C}_i)) - \mathbf{D}(\mathbf{t}_i, \text{mean}(\mathcal{C}_i \cup \{\mathbf{r}\})))$
- 10: Remove \mathbf{r} from partition h and update $\text{mean}(\mathcal{C}_h)$
- 11: Add \mathbf{r} to partition j and update $\text{mean}(\mathcal{C}_j)$
- 12: Return \mathcal{C}

2.3 Solving the GUIDED TEAM-PARTITIONING Problem

Let's assume we are given k partitions (such that mean of each partition \mathcal{C}_i is close to \mathbf{t}_i) and along with each partition, we are given q points (for all $q = 1 \dots \ell$) to be removed from that partition. Now we can develop a dynamic programming algorithm to optimally identify ℓ points to be removed from all partitions. Let $B(i, q)$ denote the benefit of removing the given q points from the i^{th} partition. The benefit is defined as how much the mean of a partition will get closer to its designated target. More formally the benefit of removing points \mathcal{R}_q from partition \mathcal{C}_i with target \mathbf{t}_i is $\mathbf{D}(\text{mean}(\mathcal{C}_i), \mathbf{t}_i) - \mathbf{D}(\text{mean}(\mathcal{C}_i \setminus \mathcal{R}_q), \mathbf{t}_i)$. Let also $MBR(i-1, j-q)$ denote the benefit of removing $j-q$ points from partitions 1^{st} to $(i-1)^{\text{th}}$ partitions. The final goal is to find the $MBR(k, \ell)$, the benefit of optimally removing ℓ points from first to last (k^{th}) partition. The following dynamic programming shows how we decide upon removing points from partitions optimally.

$$MBR(i, j) = \max_{0 \leq q \leq j} \{MBR(i-1, j-q) + B(i, q)\}$$

At this stage, we have the tool to remove ℓ points from all partitions. Now we can use Algorithm 2 to find the partitions \mathcal{C} and using Algorithm 1, we can find the q points to be removed from each partition. Putting everything together, we end up with Algorithm 3 which is an efficient solution to the GUIDED TEAM-PARTITIONING problem. We call this algorithm GUIDEDSPLIT.

3. EXPERIMENTS

Algorithm 3 The GUIDEDSPLIT Algorithm

Input: Input set \mathcal{R} , target vectors \mathbf{T} , number of partitions k and number of points to be removed ℓ
Output: Partitions \mathcal{C}

- 1: $\mathcal{C} =$ Partition \mathcal{R} into k parts using Algorithm 2
- 2: **for** \mathcal{C}_i in \mathcal{C} **do**
- 3: **for** $j = 1 \dots \ell$ **do**
- 4: $B(i, j) =$ Benefit of removing j points from partition p_i using Algorithm 1
- 5: **for** $i = 1 \dots k$ **do**
- 6: **for** $j = 1 \dots \ell$ **do**
- 7: $MBR(i, j) = \max_{0 \leq q \leq j} \{MBR(i-1, j-q) + B(i, q)\}$
- 8: return $MBR(k, \ell)$ and remove corresponding ℓ from partitions

In this section, we evaluate our algorithmic solution to the GUIDED TEAM-PARTITIONING problem. **Our datasets and implementation are immediately and freely available for download.**¹. We refer the interested readers to the long version of this paper description of the datasets and baseline algorithms, and detailed description of each experiment [6].

3.1 Experimental Evaluation

Having all the datasets and target vectors ready as the input of Problem 3, we compare the efficacy of GUIDEDSPLIT algorithm with the baseline algorithms. We implemented our algorithm in Matlab and used CVX package to solve the convex optimization problem in the CONVEXOPT algorithm. We compare the cost of our algorithm compare to all other baselines. This cost refers to the distance from the mean of each partition to its target vector, see equation 1. What follows is the detailed description of each experiment.

3.1.1 Effectiveness with respect to number of discarded points ℓ

The goal of this experiment is to demonstrate how different algorithms behave with respect to the number of removed points ℓ , as presented in the definition of Problem 3.

Figure 2 shows the cost (in log scale) versus ℓ . We observe that our GUIDEDSPLIT algorithm consistently outperforms the baseline algorithms for all target vectors, and the cost of GUIDEDSPLIT is significantly less than all the baselines. We also observe when all the targets are set to the mean, the RANDOM algorithm is doing well. This is simply because a random sample of the data is expected to have the same mean as the whole data.

An interesting observation is that when targets are equal to the mean of the dataset, Figure 2(a), as ℓ increases the cost of GUIDEDSPLIT increases as well. The reason is after finding partitions when $\ell = 0$, GUIDEDSPLIT finds near perfect solution in which mean of each partition is very close to its target. Removing points from the partition discomposes the obtained solution and makes the mean of partition far from its target. We also tried this experiment on other datasets; the results were similar, and GUIDEDSPLIT outperforms in those experiments as well. However, the algorithms were more competitive on Skills dataset. Thus due to lack of space, we only report results on Skills dataset.

3.1.2 Effectiveness with respect to number of partitions k

¹<https://github.com/sanazb/TeamPartitioning>

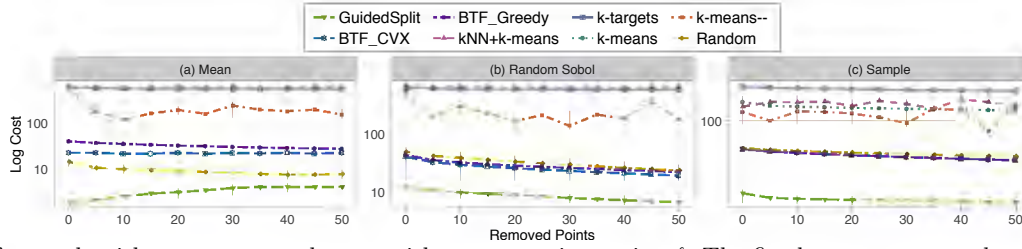


Figure 2: Different algorithms on **Skills** dataset with respect to increasing ℓ . The fixed parameters are $k = 5$, $n = 500$ and $d = 10$.

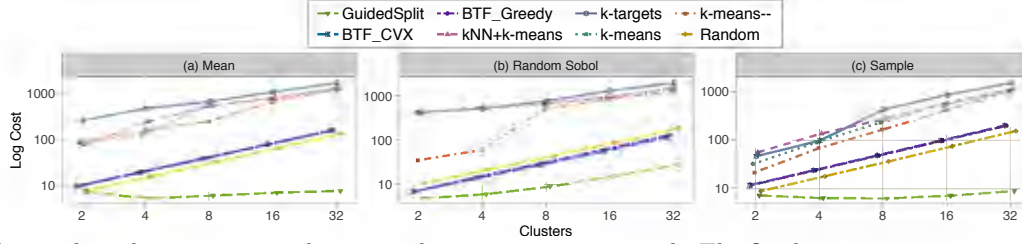


Figure 3: Different algorithms on **Skills** dataset with respect to increasing k . The fixed parameters are $n = 500$, $\ell = 50$ and $d = 10$.

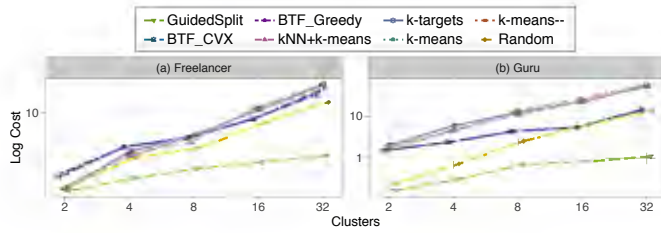


Figure 4: Different algorithms on **Freelancer**, and **Guru** datasets.

The goal of this experiment is to demonstrate how the algorithms behave with respect to the number of partitions k . In these experiments, we show the cost (in log scale) versus different number of partitions ($k = 2, 4, 8, 16, 32$) and we fixed $n = 500$, $\ell = 50$ and $d = 10$ on **Skills** dataset and $n = 502$, $d = 4$ and $\ell = 50$ for **BIA** dataset. In the experiment on **BIA** dataset, to form fair teams of students with an equal level of expertise, the targets of partitions are set to be equal to the mean of the whole dataset. We present the results in Figure 3(a), Figure 3(b), and Figure 3(c) for all three different methods of choosing the target vectors (Random-Sobol, Mean, and Sampling methods) on **Skills** dataset and Figure 5 depicts the results on **BIA** dataset.

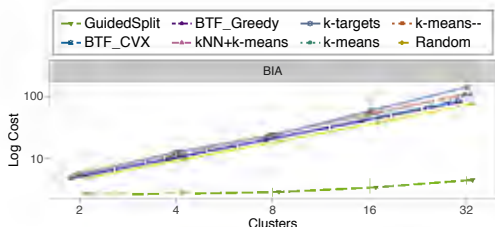


Figure 5: Different algorithms on **BIA** dataset with respect to increasing k . The fixed parameters are $n = 502$, $\ell = 50$, and $d = 4$.

3.1.3 Results on the Freelancer and Guru datasets

Guru and **Freelancer** datasets have a different dynamic compare to other datasets. In these datasets, we can use the required skills of the projects as the target vectors. Besides,

the projects posted in **freelancer** and **guru**, are the tasks that require a few skills and they can also be completed by a small team of experts. So we conducted an experiment specifically for these two datasets which indicates the performance of our algorithm with respect to the number of points n , the number of partitions k , and the number of points to remove ℓ .

For this experiment, we used the set of experts as input set \mathcal{R} , and the projects as target vectors \mathbf{T} . We tried $k = 2, 4, 8, 16, 32$, and for each value of k , we used k projects at random. We also set $n = k * 5$ and $\ell = k$ (and selected $n + \ell$ experts as random as the input \mathcal{R}). The intuition behind is due to the inherent nature of projects posted in **Guru** and **Freelancer**; these projects are small projects that usually do not need more than 5 people to be completed. By setting $n = k * 5$ and $\ell = k$, ideally, the size of each team would be 4. The results in Figure 4(a) and 4(b) illustrate **GUIDEDSPLIT** algorithm outperforms baseline algorithms on **Freelancer** and **Guru** datasets, and find teams of experts whose proficiencies are close to required skills of projects.

4. REFERENCES

- [1] R. Agrawal, B. Golshan, and E. Terzi. Forming beneficial teams of students in massive online classes. In *Learning@ scale conference*, 2014.
- [2] R. Agrawal, B. Golshan, and E. Terzi. Grouping students in educational settings. In *SIGKDD*, 2014.
- [3] S. Bahargam, D. Erdos, A. Bestavros, and E. Terzi. Personalized education; solving a group formation and scheduling problem for educational content. In *EDM*, 2015.
- [4] S. Bahargam, D. Erdös, A. Bestavros, and E. Terzi. Team formation for scheduling educational material in massive online classes. *arXiv preprint*, 2017.
- [5] S. Bahargam, B. Golshan, T. Lappas, and E. Terzi. A team-formation algorithm for faultline minimization. *Expert Systems with Applications*, 2019.
- [6] S. Bahargam, T. Lappas, and E. Terzi. The guided team-partitioning problem: Definition, complexity, and algorithm. *arXiv preprint*, 2019.
- [7] B. Golshan, T. Lappas, and E. Terzi. Profit-maximizing cluster hires. In *SIGKDD*, 2014.

Machine-Learned or Expert-Engineered Features? Exploring Feature Engineering Methods in Detectors of Student Behavior and Affect

Anthony F. Botelho
Worcester Polytechnic Institute
Worcester, MA 01605
abotelho@wpi.edu

Ryan S. Baker
University of Pennsylvania
Philadelphia, PA 19104
ryanshaunbaker@gmail.com

Neil T. Heffernan
Worcester Polytechnic Institute
Worcester, MA 01605
nth@wpi.edu

ABSTRACT

There is a long history of research on the development of models to detect and study student behavior and affect. Developing computer-based models has allowed the study of learning constructs at fine levels of granularity and over long periods of time. For many years, these models were developed using features based on previous educational research from the raw log data. More recently, however, the application of deep learning models has often skipped this feature-engineering step by allowing the algorithm to learn features from the fine-grained raw log data. As many of these deep learning models have led to promising results, researchers have asked which situations may lead to machine-learned features performing better than expert-generated features. This work addresses this question by comparing the use of machine-learned and expert-engineered features for three previously-developed models of student affect, off-task behavior, and gaming the system. In addition, we propose a third feature-engineering method that combines expert features with machine learning to explore the strengths and weaknesses of these approaches to build detectors of student affect and unproductive behaviors.

Keywords

feature engineering, student affect, disengaged behavior, deep learning

1. INTRODUCTION

The educational data mining community has developed numerous models to detect unproductive student behaviors and affective states and study how these measures correlate with short- and long-term learning outcomes. Estimates produced by detectors of student affective states and unproductive behavior, for example, have been found to predict student standardized test scores [15], whether a student chooses to attend college [16], and whether they pursue a degree in STEM [17], and even later pursue a STEM career

[11], from estimates produced from interaction logs collected as they worked on mathematics problems in seventh grade. The predictive power of these detectors along with a general desire to understand and improve the student learning process has led to a significant amount of research around developing these models.

The primary goal of this paper is to compare the two methods of generating features to predict student affect and unproductive behaviors. It is also important to consider whether some models perform better for certain types of features. Conversely, other simpler models such as a decision tree or logistic regression require feature engineering or aggregation in order to incorporate labeled and unlabeled data. Additionally, these models would be unable to easily observe unlabeled data in a semi-supervised learning manner [19].

In this work we re-develop detectors of student affective state [5], off-task behavior [15], and gaming the system [13], comparing new models to previously-developed models, to address the following research questions: 1) Which leads to better model performance (AUC ROC and Kappa), expert-engineered features or machine-learned features, for detectors of affect and unproductive behaviors? 2) Does combining expert-engineered features and machine learning-based feature generation improve model performance for detectors of affect and unproductive behaviors? 3) Does incorporating unlabeled data improve model performance for detectors of affect and unproductive behaviors?

The development of affect detectors within ASSISTments has improved in recent years. Initial work has explored the use of expert-engineered features to develop and evaluate detectors through a population validity study [12]. Recently, a deep learning approach was applied [5], utilizing the engineered features within a recurrent neural network to predict the four labels of affective state simultaneously over time.

This work utilizes two datasets¹ consisting of student interaction log data collected within the ASSISTments computer-based learning platform. The content within the system consists primarily of mathematics problems for students in grades 6-8. Students working in the system receive immediate correctness feedback on each problem with the ability to make multiple attempts, and have the ability to ask for on-

Anthony F. Botelho, Ryan S. Baker and Neil T. Heffernan
"Machine-Learned or Expert-Engineered Features? Exploring Feature Engineering Methods in Detectors of Disengaged Behavior and Affect" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 508 - 511

¹All data used in this work is made openly available at the following link: http://tiny.cc/ML_or_Expert

demand computer-provided aid in the form of hints or scaffolded problems. These interactions and timing information are the data used to construct most of the expert-engineered features utilized in this work. The ground-truth labels of both student affect and off-task behavior were collected using quantitative field observations following the Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP) [1].

We analyze gaming the system using a data set collected via text replays [2]. The ground-truth labels of student gaming were collected using post-hoc examinations of sequences of student data following a set of criteria outlined in [2].

2. METHODOLOGY

Jiang et al. [10] compared two feature engineering methods, expert-generated and deep learning-based, for the development of affect and off-task behavior detectors within the Betty's Brain learning system. This work will test the generalizability of their findings, explore the strengths and weaknesses of each method, and explore the incorporation of unlabeled data during model training.

Each of the models described next were trained and evaluated using stratified 10-fold student-level cross validation. We stratified each fold based on the number of occurrences of positive labels of each outcome label at the student level to generate the folds of the cross validation.

2.1 Utilizing Expert-Engineered Features

The first set of models use expert-engineered features to detect each label of student affect, off-task behavior, and gaming the system using methods similar to those implemented in previous works. The expert features are first generated using the raw action-level log data; 92 features are created for the affect and off-task data while 33 are generated for the gaming detectors in alignment with the features used in previous works. The features are generated to describe the actions that occur in 20 seconds of observation but also include neighboring actions that go beyond those 20 seconds to capture the full context. We apply a Naive Bayes classifier, a REP tree classifier and a Long-Short Term Memory (LSTM) deep learning network [9] for the gaming, off-task behavior, and affect detection tasks as these were found to work well in previous works [13][15][5].

2.2 Deep Learning Models

Unlike the expert-engineered features, the machine learned feature set uses the raw action logs of each student, ignoring the clips and clip-level features described in the previous section. For this feature set, a LSTM model is applied over the raw data to predict each outcome using a set of uninterpretable features learned within the hidden layer of the network. One potential drawback of using a LSTM model in this way is that it assumes that each timestep in the given sequence occurs at regular intervals which, of course, is not the case. Therefore, to reduce the variance of this interval, a similar practice as was applied to the affect detector model using expert-engineered features. This allows the model to divide sequences of student actions where long intervals may occur between subsequent actions; where the amount of time between two subsequent actions of the same student is greater than 5 minutes, the sequence is divided into two smaller sequences to be input into the model.

Each model utilized the same raw action-level log data that was used to generate the expert-engineered features described in the previous section. In addition to the interaction descriptors such as response correctness and whether the student requested a hint, the knowledge component associated with each problem was also included as a large 1-hot encoded vector in an effort to supply these LSTM models with the same information with which the expert-generated features had access. In addition to these described action logs, the set of features supplied to the gaming model included an additional field corresponding to the computed Levenshtein distance of each students sequence of incorrect responses (where such sequences of incorrect responses existed) within each problem as was computed for the expert-engineered features of this detector. We incorporated this feature to provide consistency in the information that is exposed to both the machine learning model and the expert feature-engineering process, although we acknowledge that the feature itself is a transformation of the raw responses (i.e. can be described as an expert feature) and was only found to be predictive of gaming the system through prior work exploring the development of expert features for this task [14].

Each of the three LSTM models created for each label of student affect, off-task behavior, and gaming the system followed the same general structure comprised of an input layer feeding into a fully-connected recurrent hidden layer of 200 LSTM nodes, and then feeding into an output layer of either 2 nodes (corresponding to a 1-hot encoded positive and negative indicator of either off-task behavior or gaming the system) or 4 nodes (corresponding to a 1-hot encoded vector with one value per observed affective state). The purpose of the hidden layer is to learn a set of 200 features from the raw action logs that are predictive of each outcome label. The commonly applied technique of dropout [18] is applied between the hidden and output layers of the network in an attempt to reduce overfitting. In all cases, a softmax activation function is applied to the output of each model and trained using multiclass cross entropy.

2.3 Machine-Learned Expert-Inspired Features

The third feature set proposed for comparison combines aspects of both expert-engineered features and machine learning. Expert features may be able to help guide a machine learning model to learn better sets of features than either method individually. In addition, since each set of expert features were presumably developed with a particular set of outcome measures in mind, such labels may also be able to help guide a machine learning model to produce meaningful, albeit uninterpretable, sets of features to detect such behaviors and affective states.

Specifically, this method utilizes a 2-step training process for a machine learning model. First, an LSTM model is built to use the raw action-level logs as input (just as was done in the previous section for the models utilizing machine learned features), but in addition to predicting each label, the model is trained to predict the set of expert-engineered features as a multi-task learning problem [7]. As the affect and off-task behavior detectors utilize the same set of action logs and expert-engineered features, we build one model to read the interaction logs. This model will predict each of the set of

Table 1: Comparison of feature sets across each of the detector models with and without co-training.

Outcome	Feature Set	Model	AUC	Kappa	Co-Trained* AUC	Co-Trained* Kappa
Off-Task	Expert	REP Tree	.734	.352	.796	.369
	Machine Learned*	LSTM	.657	.073	.657	.073
	Machine Learned Expert	REP Tree	.753	.400	.781	.405
Gaming	Expert	Naive Bayes	.774	.362	.856	.180
	Machine Learned*	LSTM	.542	-.005	.542	-.005
	Machine Learned Expert	Naive Bayes	.774	.290	.847	.327
Affect (Collectively)	Expert	LSTM	.760	.172	.777	.112
	Machine Learned*	LSTM	.695	.041	.695	.041
	Machine Learned Expert	LSTM	.662	.043	.607	.037
Confusion	Expert	LSTM	.730	.042	.762	.059
	Machine Learned*	LSTM	.666	.042	.666	.042
	Machine Learned Expert	LSTM	.609	.01	.596	.018
Engaged Concentration	Expert	LSTM	.775	.281	.791	.289
	Machine Learned*	LSTM	.713	.210	.713	.210
	Machine Learned Expert	LSTM	.671	.188	.611	.090
Boredom	Expert	LSTM	.775	.148	.783	.178
	Machine Learned*	LSTM	.690	.137	.690	.137
	Machine Learned Expert	LSTM	.677	.041	.613	.005
Frustration	Expert	LSTM	.761	.054	.772	.050
	Machine Learned*	LSTM	.713	.060	.713	.060
	Machine Learned Expert	LSTM	.689	.019	.609	.026

*All co-training used an LSTM model. Models using the machine learned features all used co-training.

expert-engineered features corresponding with the given set of actions, the affective state label, and the off-task behavior label simultaneously. Similarly, for the gaming detector, the raw actions are supplied as input to a LSTM model that predicts both the set of expert-engineered features and the gaming labels. In this way, the hidden layer of the respective models is regularized to learn a set of features that is both able to construct the set of expert features (although likely with some error) as well as predict the outcome labels for which the features are intended.

Once these LSTM models are trained, the hidden layer is extracted and used as the third and final set of features compared in this work. This feature set, referred to as “machine-learned expert-inspired” features, is then supplied as input to each of the respective models used in previous work.

2.4 Exploring the use of Co-Training

In this work, we use the term “co-training” to describe the use of both labeled and unlabeled data to inform model estimates and the methods differ from that of other works describing co-trained models [3]. Given the nature of the observation-based label collection procedure, not all examples in our data (whether considering actions or clips) has an associated affect or behavior label. While there are several modeling methods that exist to incorporate this unlabeled data into each model, the already-described LSTM model inherently allows for this co-training to occur given its sequential structure. The model uses the current supplied timestep along with a learned-aggregation of previous time steps in order to better inform each prediction.

3. RESULTS AND DISCUSSION

We compare the results of each set of models within each of the three outcome measures of affect, off-task behavior,

and gaming behavior using the metrics of AUC ROC and Cohen’s Kappa. In the case of affect, AUC ROC is calculated using a multi-class variant of the metric [8], while Kappa is calculated as multi-class Cohen’s Kappa, while the models of off-task behavior and gaming use an optimized form of Kappa by learning an optimal decision (0,1) threshold using the training set of each respective fold within the cross validation. Higher values of either metric are indicative of higher model performance with AUC values at 0.5 and Kappa values at 0 indicating chance performance.

The results of each model is reported in Table 1. Compared to the models utilizing machine learned features, the expert-engineered features lead to notably higher performance across all the outcome labels in regard to both AUC and Kappa. When comparing the performance of the models using the machine-learned expert features (our proposed third feature set), the difference in performance is not as dramatic, but leans in favor of the expert-engineered features leading to superior models. By contrast, the machine learned expert features led to models that outperform those using expert-engineered features in regard to off-task behavior in terms of both AUC and kappa and is equal in regard to detecting gaming in terms of AUC, but appears to perform less well in detecting student affect.

Despite the small number of cases where the models trained from expert-engineered features did not outperform the others in either AUC or Kappa, these models exhibited consistently high performance in both metrics across all outcomes. It can further be concluded that co-training led to higher AUC than the non-co-trained variant of each detector. This, however, was not always the case in regard to Kappa, where the co-trained models of gaming and affect using expert-engineered features exhibited notably lower values.

4. FUTURE WORK

Among the detectors, it was found in some cases that there was disagreement between the metrics used; the co-trained model of affect exhibited higher AUC than the non-co-trained model, but a lower value of Kappa. Previous work has explored this case across several commonly-applied metrics [4], but further work is needed to further explore and leverage modeling techniques to produce detectors that exhibit high performance across all these metrics.

The use of the highest-performing models across each of these detectors can also be used in future research to study other aspects of student learning. This extends beyond the already-discussed application in predicting student longer-term outcomes, and includes the study of other aspects such as the dynamics and chronometry, studied using affect detectors in prior work [6].

5. CONCLUSIONS

This work investigates whether expert-engineered features lead to higher performing detectors of student affect and disengaged behavior as compared to using automatically-distilled features learned through a machine-learning approach. We found that the use of expert features led to the most consistently high-performing models. Using co-training led to even better models in most cases.

The use of expert-engineering to develop features, while perhaps more difficult in regard to time, effort, and likely cost, does appear to lead to greater benefits than simply applying a machine learning model to automatically distill features from the raw data, based on the results found in this work.

6. ACKNOWLEDGMENTS

We thank multiple current NSF grants (ACI-1440753, DRL-1252297, DRL-1109483, DRL-1316736, DGE-1535428, OAC-1724889, OAC-1636782 & DRL-1031398), the US Dept. of Ed (IES R305A180401, R305A120125 & R305C100024 and GAANN), and the ONR.

7. REFERENCES

- [1] R. Baker, J. Ocumpaugh, and J. Andres. Brompt quantitative field observations: A review. *R. Feldman (Ed.) Learning Science: Theory, Research, and Practice*, In Press.
- [2] R. S. Baker, A. T. Corbett, and A. Z. Wagner. Human classification of low-fidelity replays of student actions. In *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*, volume 2002, pages 29–36, 2006.
- [3] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.
- [4] N. Bosch and L. Paquette. Metrics for discrete student models: Chance levels, comparisons, and use cases. *Journal of Learning Analytics*, 5(2):86–104, 2018.
- [5] A. F. Botelho, R. S. Baker, and N. T. Heffernan. Improving sensor-free affect detection using deep learning. In *International Conference on Artificial Intelligence in Education*, pages 40–51. Springer, 2017.
- [6] A. F. Botelho, R. S. Baker, J. Ocumpaugh, and N. T. Heffernan. Studying affect dynamics and chronometry using sensor-free detectors. pages 157–166, 2018.
- [7] R. Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.
- [8] D. J. Hand and R. J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning*, 45(2):171–186, 2001.
- [9] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [10] Y. Jiang, N. Bosch, R. S. Baker, L. Paquette, J. Ocumpaugh, J. M. A. L. Andres, A. L. Moore, and G. Biswas. Expert feature-engineering vs. deep neural networks: Which is better for sensor-free affect detection? In *International Conference on Artificial Intelligence in Education*, pages 198–211. Springer, 2018.
- [11] J. Makhoulf and T. Mine. Predicting if students will pursue a stem career using school-aggregated data from their usage of an intelligent tutoring system. In *Educational Data Mining 2018*, pages 533–536, 2018.
- [12] J. Ocumpaugh, R. Baker, S. Gowda, N. Heffernan, and C. Heffernan. Population validity for educational data mining models: A case study in affect detection. *British Journal of Educational Technology*, 45(3):487–501, 2014.
- [13] L. Paquette, R. S. Baker, A. de Carvalho, and J. Ocumpaugh. Cross-system transfer of machine learned and knowledge engineered models of gaming the system. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 183–194. Springer, 2015.
- [14] L. Paquette, A. de Carvahlo, R. Baker, and J. Ocumpaugh. Reengineering the feature distillation process: A case study in detection of gaming the system. In *Educational data mining 2014*. Citeseer, 2014.
- [15] Z. A. Pardos, R. S. Baker, M. O. San Pedro, S. M. Gowda, and S. M. Gowda. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, pages 117–124. ACM, 2013.
- [16] M. O. Pedro, R. Baker, A. Bowers, and N. Heffernan. Predicting college enrollment from student interaction with an intelligent tutoring system in middle school. In *Educational Data Mining 2013*, 2013.
- [17] M. O. San Pedro, J. Ocumpaugh, R. S. Baker, and N. T. Heffernan. Predicting stem and non-stem college major enrollment from middle school interaction with mathematics educational software. In *Educational Data Mining 2014*, pages 276–279, 2014.
- [18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
- [19] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

Shedding Light on the Automated Essay Scoring Process

David Boulanger
Athabasca University
Edmonton, AB, Canada
dboulanger@athabascau.ca

Vivekanandan Kumar
Athabasca University
Edmonton, AB, Canada
vivek@athabascau.ca

ABSTRACT

This paper explores in depth the suitability of the 2012 Automated Student Assessment Prize (ASAP) contest's essay datasets. It evaluates the potential of deep learning and state-of-the-art NLP tools in automated essay scoring (AES) to predict not only holistic scores but also the finer-grained rubric scores, an area underexplored but essential to provision formative feedback and uncover the AI reasoning behind AES. For comparison purpose, this paper advocates the need for transparency when sharing AES processes and outcomes. Finally, it reveals the insufficiency of ASAP essay datasets to train generalizable AES models by examining the distributions of holistic and rubric scores. Findings show that the strength of agreement between human and machine graders on holistic scores does not translate into similar strength on rubric scores and that the learning made by the machine barely exceeds the performance of a naïve predictor.

Keywords

Automated Essay Scoring; Automated Student Assessment Prize; Deep Learning; Natural Language Processing; Rubrics

1. INTRODUCTION

With the soaring development of deep learning and NLP, the interest in pushing further the limits of AES and harnessing the potential of valuable hand-graded essay datasets has been reinvigorated. This effervescence, however, is accompanied by a lack of transparency, although involuntary, that sometimes even prevent to measure and compare progress made in AES, prompting for some protocol on how to report research processes and outcomes in this area. This paper assesses the value of one of the most used set of datasets to train AES models, the free Automated Student Assessment Prize datasets, and sheds light on the factors that alter proper comparison between researchers' models. It investigates the capacity of deep learning in pair with the extensive range of writing features available nowadays from cutting-edge NLP tools to predict both holistic and rubric scores, a research orientation barely explored at this time and key for the provision of finer-grained formative feedback to student writers.

2. METHODOLOGY

The Automated Student Assessment Prize (ASAP) contest was launched in 2012 to evaluate the progress in AES and its readiness to be implemented across the United States in state-wide

writing assessments [4]. The Hewlett-Packard Foundation funded the Automated Student Assessment Prize contest that was hosted on the Kaggle platform to invite as many participants from the community of data scientists. Eight essay datasets were collected from these state-wide assessments and were graded following a thorough scoring process by subcontracted commercial vendors. These state-wide assessments were written by Grade 7-10 students from six different states. Each essay dataset originated from a single state-wide assessment targeting a specific grade (7-10) and state. The ASAP contest required participants to develop a model that would automatically score all these datasets and measure the level of agreement between their developed machine grader and the resolved human scores using the quadratic weighted kappa (QWK) metric. Both commercial vendors and data scientists from academia participated in the contest, and winners were determined based on the average QWK on all eight essay datasets. Although necessary for contest purposes, this proves not to be an effective way to transparently compare and report research processes and results given that each essay dataset has a unique setting, such as the type of writing construct (persuasive, source-based, expository, narrative), which at its turn requires a customized approach. Following the publication of the contest outcomes [4], warnings [4] came against concluding that AES was actually beating human graders just because it merely surpassed the level of agreement among human graders. For example, it was showcased [4] that AES systems could be fooled by only submitting text with the proper number of words, demonstrating "how smart" AES really was. Obviously, the ASAP study design had some limitations including the fact that none of the essay datasets had an articulated writing construct, most essays were too short and had on average between 94 and 381 words, suffered from a bias in the way holistic scores were resolved, had small scoring scales, and most of them did not have any scoring rubric [4]. The essay dataset #8 (Grade 10, narrative, mean number of words of 622) was the only exception to this and could at least test the writing ability of students. This makes the eighth dataset promising for machine to learn and provision formative feedback to both students and teachers. However, the main shortcoming of the eighth dataset is its small sample size, that is, only 722 of the original essay samples have been made available to the public.

This paper continues and extends the work undertaken by [1] in which the potential of deep learning in AES is evaluated. Because the essay sample size is very small, the latest advances in NLP along with a deep neural network (at least three hidden layers) were adopted to facilitate the detection of writing patterns instead of working directly with the raw text of student writings through a sequence model (i.e., a long short-term memory recurrent neural network). Hence, each of the 722 essays was processed by the Suite of Automatic Linguistic Analysis Tools¹ (SALAT). Among others, SALAT provides a set of baseline metrics related to word

David Boulanger and Vivekanandan Kumar "Shedding Light on the Automated Essay Scoring Process" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 512 - 515

¹ <https://www.linguisticanalysistools.org/>

frequency, text content, and lexical sophistication. The generic lexical sophistication metrics include among others psycholinguistic word information metrics (e.g., concreteness and familiarity), lexical frequency and range metrics (words occurring in a wider range of texts) based on large English corpora, and syntactic categories (e.g., number of adjectives and nouns). It also measures over 400 indices of lexical sophistication related to word and n-gram frequency and range, academic language, n-gram strength of association, contextual distinctiveness, word recognition norms, semantic network, and word neighbors. Several of these metrics are normed according to the number of word or n-gram occurrences found in large corpora of English writings and frequency lists. Moreover, SALAT supplies 367 indices evaluating the syntactic sophistication and complexity of English writing, grouped into four categories: 14 syntactic complexity analysis indices, 31 indices on clausal complexity, 132 indices related to phrasal complexity, and 190 syntactic sophistication indices. Finally, SALAT comes up with a set of over 150 features related to local, global, and overall text cohesion. Texts are first lemmatized and grouped per sentence and paragraph before computing any cohesion metric. It also employs a part-of-speech tagger and synonym sets from the WordNet lexical database for the computation of these cohesion metrics, which are grouped among five categories: connectives, givenness, type token ratio, lexical overlap, and semantic overlap.

Hence, a vector of 1,463 writing features was generated for each essay sample. Because of this huge number of features, [1] computed the correlation coefficients using the non-parametric Spearman correlation between every writing metric and holistic scores. The writing metrics were then sorted by the strength of association from strongest to weakest. Several subsets of writing metrics containing the most correlated metrics were created, that is, vectors with 192, 128, 96, and 64 writing features by essay sample were input into a deep neural network. The best results were delivered through the 96-feature dataset², which is re-used in this study for rubric score prediction and for comparison with [1].

ASAP eighth essay dataset encloses holistic scores resolved by adjudication rules applied on the rubric scores of two human graders. Each human grader is asked to score the essay as per six rubrics and to assign a discrete number between 1 and 6 to each rubric. However, holistic scores are derived from only four rubric scores (1, 2, 5, and 6) according to the following formula: $R1 + R2 + R5 + 2 \cdot R6$. Each rubric score is the sum of the two human ratings except for cases in which the human ratings do not consistently agree with each other. In such cases, a third human rater scores the essay, assigning 1-6 scores to each rubric. The rubric scores are then calculated by multiplying the third human rater's ratings by two. This involves that all rubric scores range from 2 to 12. However, ASAP only provides individual human ratings (1-6). Hence, this research work compiled those resolved rubric scores as described above.

3. ANALYSIS & RESULTS

The purpose of this study is to investigate the feasibility and the benefits of applying automated essay scoring at the rubric level. Rubric scores are a form of high-level formative feedback that can be provided both to student writers and teachers. Moreover, as depicted in Figure 1 [1], accurately predicting essays' holistic scores on a wide scale (i.e., 10-60) when trained with a rather

small sample dataset and an unbalanced score distribution proves a daunting task that goes far beyond computing a good "kappa" value, as publications subsequent to the ASAP contest continued to report. 530 (73.4%) essay scores out of 722 range between 30 and 40 (40-60%), while 707 (97.9%) essay scores lie between 25 and 50 (30-80%), leaving to the machine only three essays from which to learn high-quality writing (> 80%).

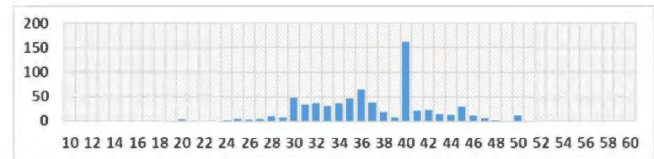


Figure 1. Distribution of holistic scores.

This paper hypothesizes that holistic scores would be better predicted through the prediction of its constituent rubric scores. For instance, if the distribution of holistic scores was uniform, there would be about $722/51=14$ examples per holistic score, not much to learn all the intricacies of English writing. On the other side, the scale of rubric scores ranges from 2 to 12, implying that in the best case there would be 66 essays per rubric score, more essays from which to assess a narrower writing competence. Figure 2 exhibits the distribution of scores of each rubric.

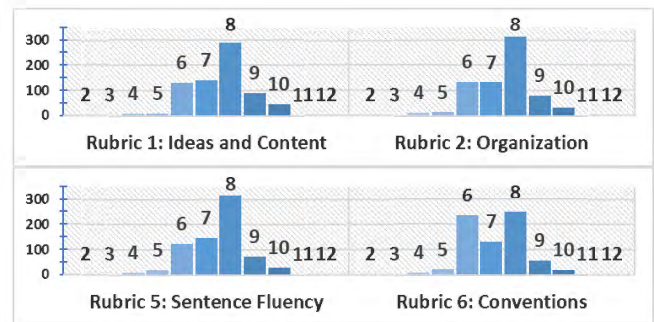


Figure 2. Distributions of rubric scores.

Table 1 delineates the agreement levels among rubrics and shows that they generally strongly agree with each other. It can also be noticed from Figure 2 that the most frequent score within each rubric is "8," suggesting that an AES system could naively predict scores of "8" for all rubrics and a holistic score of 40. The performance of such a naïve AES system (aka majority classifier) is measured by three metrics and reported in Table 2: the QWK, the percentage of exact predictions, and the percentage of adjacent predictions (off by 1). The QWK measures the level of agreement between two raters by controlling for random guessing and by penalizing exponentially (quadratic) higher distances between pairs of ratings.

Both supervised classification and regression techniques have been leveraged to model rubric scores. Table 3 reports the performance of six distinct deep learning architectures trained,

Table 1. Level of agreement among rubrics (QWK)

	R1	R2	R5	R6	Avg.
Rubric 1	-	0.88	0.79	0.68	0.78
Rubric 2	0.88	-	0.80	0.71	0.80
Rubric 5	0.79	0.80	-	0.79	0.79
Rubric 6	0.68	0.71	0.79	-	0.73

² Project hosted on <https://github.com/davidbo57/edm2019>

Table 2. Performance of a naïve AES system

	R1	R2	R5	R6	Avg.	HS
QWK	0	0	0	0	0	0
Exact %	40.0	43.1	43.8	34.3	40.3	22.3
Adj. %	72.0	72.3	74.1	60.0	69.6	26.5

Avg.=Average of all rubrics combined; HS=Holistic scores

validated, and tested on the ASAP eighth dataset and its derived 96-feature dataset [1]. The 722 essays have been randomly split into a training set (85% = 614 essays) and testing set (15% = 108 essays). Although validation results are not reported here, 5-fold cross-validation was performed for each architecture, implying that the training set was randomly split into a smaller training set (80% = 491 essays) and a validation set (20% = 123 essays). Furthermore, this study unsuccessfully attempted to replicate the results reported by [1], that is, a QWK of 0.80 by running 10 times the available code. The obtained QWK values ranged between 0.51 and 0.74 with a mean of 0.63, suggesting that the results reported by [1] were subject to sampling error. Because this shuffling of essays among the training, validation, and testing sets combined with an unbalanced distribution of essay scores highly influence the machine learning and the resulting predictions, this study reports performance on the training and testing sets as the average of five replications [2] (different samplings) instead of picking up the “highest kappa” generated to avoid overfitting the AES models to the testing set.

The first architecture is a classifier selecting the most likely rubric score among a set of 11 discrete scores between 2 and 12. The second architecture is identical except that it leverages in addition a bagging ensemble technique. In other words, each model trained per fold during cross-validation makes up a machine grader, giving five machine graders, each with its own “expertise,” that determine the final rubric score by averaging their predicted score. The third architecture is a regressor that computes a single real-number score (e.g., 7.4398) and rounds it to the nearest integer, truncating it to 2 or 12 if the real number falls short or exceeds the scale. The fourth architecture is a regression ensemble (bagging). The fifth architecture is a regressor that considers the interdependencies among rubrics, from which holistic scores are derived. Thus, instead of predicting a single rubric score, this approach predicts all four rubric scores at once. Then the agreement level between the machine and the human graders for each rubric is analyzed and reported independently. Finally, the sixth architecture employs a bagging ensemble technique on top of the fifth architecture to predict rubric scores. Table 4 delineates the final hyperparameters of the various architectures.

4. DISCUSSION & CONCLUSION

Figure 2 shows that the most frequent score is “8” for all rubrics and that all rubrics have similar distributions except for Rubric 6. In addition, Table 1 shows that the rubric scores are highly dependent on each other, that is, they exhibit consistent agreement among them. The QWK values all range from 0.68 to 0.88. In particular, Rubric 6 has a slightly lower average agreement level with other rubrics (QWK=0.73) than the other rubrics (R1=0.78, R2=0.80, R5=0.79), which may account for its slightly different distribution. In practical terms, Rubric 6 being about grammatical and spelling conventions, this may relate to the intuition that the content and ideas, organization, or the sentence fluency of a text do not entirely depend on the mechanics of the student’s writing.

Table 2 reveals that 40.0%, 43.1%, 43.8%, and 34.3% of the rubric scores have been given an “8” by the human graders. This implies that systematically assigning an “8” to all rubrics would “predict” on average exact and adjacent scores 40.3% and 69.6% of the times. An awesome performance for a completely naïve AES model! Interestingly, the QWK metric proves to be an effective indicator to detect random guessing as shown by the 0’s in every rubric. It is, therefore, imperative that trustable AES models reach exact and adjacent match percentages significantly higher than those of this majority classifier.

By aggregating and selecting the best performance per rubric on the testing set, all architectures combined as demonstrated at the bottom of Table 3, it can be observed that on average the QWK’s lie between 0.48 and 0.55, much lower than the 0.80 QWK reported in [1] and the average 0.63 QWK replicated in this study

Table 3. AES performance on training/testing set

	R1	R2	R5	R6
1. Classification				
QWK	0.61 / 0.52	0.69 / 0.52	0.66 / 0.43	0.66 / 0.49
Exact %	52.2 / 49.2	57.0 / 49.9	53.8 / 44.4	57.5 / 51.0
Adj. %	86.3 / 81.8	87.8 / 82.0	88.6 / 80.4	86.0 / 76.5
2. Classification Ensemble				
QWK	0.62 / 0.52	0.66 / 0.51	0.64 / 0.43	0.68 / 0.5
Exact %	51.4 / 47.3	55.3 / 49.0	52.9 / 43.3	56.7 / 49.7
Adj. %	88.6 / 82.8	89.3 / 83.5	89.9 / 83.1	88.6 / 78.5
3. Regression				
QWK	0.85 / 0.41	0.85 / 0.50	0.83 / 0.46	0.85 / 0.55
Exact %	64.4 / 38.7	66.2 / 40.0	63.2 / 39.8	64.6 / 42.6
Adj. %	97.6 / 83.5	98.1 / 86.2	98.4 / 85.7	98.5 / 87.7
4. Regression Ensemble				
QWK	0.81 / 0.41	0.80 / 0.50	0.80 / 0.47	0.82 / 0.55
Exact %	60.3 / 40.2	58.0 / 39.8	58.8 / 40.6	58.8 / 42.8
Adj. %	97.1 / 86.8	97.4 / 87.2	98.1 / 86.8	98.3 / 87.7
5. Multiple Regression				
QWK	0.76 / 0.52	0.77 / 0.48	0.78 / 0.48	0.76 / 0.52
Exact %	53.9 / 41.8	54.1 / 38.3	54.7 / 39.3	48.8 / 34.1
Adj. %	94.8 / 85.5	96.0 / 86.8	97.1 / 86.6	96.9 / 88.3
6. Multiple Regression Ensemble				
QWK	0.72 / 0.51	0.74 / 0.52	0.75 / 0.47	0.72 / 0.52
Exact %	51.4 / 40.0	52.2 / 39.6	51.9 / 41.3	46.4 / 33.4
Adj. %	94.0 / 86.1	95.1 / 89.5	96.4 / 86.1	96.4 / 88.6
Best Performances on Testing Set Combined				
QWK	0.52	0.52	0.48	0.55
Exact %	49.2	49.9	44.4	51.0
Adj. %	86.8	89.5	86.8	88.6

Table 4. Hyperparameters of deep learning architectures

	Classification	Regression	Multiple Regression
Layers	96-64-64-32-32-11	96-96-64-32-1	96-96-64-32-4
Reg	Dropout (0.6)	Dropout (0.3)	Dropout (0.2)
Opt	Adam	Adam	Adam
Loss	Cat. cross-entropy	MSE	MSE
Act	Elu, Softmax	Elu, None	Elu, None
Epoch	200	500	300
Batch	128	128	128
Kernel	Normal	Normal	Normal

Layers=Number of layers and number of nodes per layer; Reg=Regularization; Opt=Optimizer; Act=Activation function; Kernel=Kernel initialization

from [1]’s code, meaning that a high agreement level between human and machine graders on holistic scores does not necessarily translate into an as high level of agreement on rubric scores. This may be surprising given the much smaller scale of rubric scores. Moreover, deep learning’s best exact match percentages range between 44.4% and 51.0%, only 4-11% higher than the majority classifier (40.3%). As for adjacent match percentages, on average rubric scores fall between 86.8-89.5% of the times around +/-1 the actual rubric scores compared to 69.6% for the majority classifier. In other words, approximately 39% of rubric scores are off by 1 and 12% are off by 2 or more.

Research works having dealt with models predicting rubric scores trained on the ASAP datasets are quite rare. Among the very few is a feature-engineered AES system called SAGE [5], which was designed and tested using several machine learning techniques: linear regression, regression trees, neural network, random forest, and extremely randomized trees. The literature shows that SAGE appears to be currently the most powerful AES system in terms of agreement level (QWK=0.805) on ASAP eighth dataset. SAGE is unique in that it calculated for the first time 29 novel semantic coherence and 3 novel consistency metrics in addition to 72 more traditional linguistic and content metrics. Because these novel metrics were directly related to the second rubric (Organization), the authors also trained a model to predict the score of that rubric and reported a QWK of 0.70. However, it is not specified whether the agreement level was assessed using the 1-6 human rubric scores or the 2-12 resolved rubric scores. Interestingly, SAGE was tested using all the 1,527 [4] original essay samples available during the ASAP contest. Since only a subset (722) is now available since the contest is over, most research works have reported their performance on that subset, thus jeopardizing the fair comparison of those AES models against SAGE. Similarly, another AES system was implemented using a Common N-Gram classifier [3]. Each essay was represented by a set of n-grams, made up of either the actual words, stemmed words, or characters. The AES system trained two machine graders, both of which predicted rubric scores on the 1-6 scale. Table 5 reports the best agreement levels per rubric between the machine and human graders and between the two human graders as reported by [3].

Table 5’s results provide further evidence that high levels of agreement between machine and human graders on the ASAP eighth dataset’s holistic scores do not translate into similarly high agreement levels on the rubric scores, which again questions the humanlike “expertise” of AES systems [4]. This also leads to

conclude that the ASAP eighth dataset, the best that the ASAP contest delivered in terms of realistic essays, is clearly insufficient to train effective AES models. To counter tentatively the imbalance in the distribution of holistic scores, this study retrained the six models described earlier and assigned weights to the essay samples based on the frequency of essays per holistic score as shown in Figure 1. Yet, the performances were inferior to those reported in Table 3 and, hence, not reported.

Table 5. Best QWK results per rubric

	Rubric 1	Rubric 2	Rubric 5	Rubric 6
H1	0.482	0.455	0.489	0.454
H2	0.394	0.377	0.459	0.436
H1H2	0.523	0.533	0.498	0.532

H1=1st human grader; H2=2nd human grader; H1H2=Agreement between both graders

Finally, this paper analyzed the accuracy of the fifth and sixth models in predicting holistic scores out of the four predicted rubric scores to better explain how they are derived. The two models at best yielded on average a QWK of 0.56 and exact and adjacent match percentages of 12.5% and 26.8. This performance is equivalent or worse than the majority classifier. AES at the rubric level seems more challenging than it first appears.

For comparison purpose, this study was constrained to only analyzing the dataset of the 96 most correlated metrics with holistic scores as used by [1]. However, next steps should include identifying the writing indices that really contribute to each rubric score and assessing the size of their contribution. This would imply training each rubric model many times, each time randomly selecting a new set of writing indices (e.g., 50) and analyzing how the rubric model’s performance and the weights of its neural network’s very first layers vary as the set of writing features change. Moreover, a hybrid feature-based and end-to-end deep learning architecture should be explored, combining both writing features from NLP tools and the raw texts input into a sequence model with an attention mechanism. This would produce very fine-grained feedback by not only providing rubric scores but also highlighting portions of an essay that contributed to the predicted holistic and rubric scores and reporting the suboptimal writing indices affecting these text passages, hinting student writers about the importance and ways to improve their writing.

5. REFERENCES

- [1] Boulanger, D. and Kumar, V. 2018. Deep Learning in Automated Essay Scoring. *Intelligent Tutoring Systems*, 294–299.
- [2] Cozma, M., Butnaru, A.M. and Ionescu, R.T. 2018. Automated essay scoring with string kernels and word embeddings. *arXiv preprint arXiv:1804.07954*.
- [3] Jankowska, M., Conrad, C., Harris, J. and Kešelj, V., 2018. N-Gram Based Approach for Automatic Prediction of Essay Rubric Marks. *Advances in Artificial Intelligence*, 298–303.
- [4] Kumar, V., Fraser, S.N. and Boulanger, D. 2017. Discovering the predictive power of five baseline writing competences. *Journal of Writing Analytics*. 1, 1 (2017), 176–226.
- [5] Zupanc, K. and Bosnić, Z. 2017. Automated essay evaluation with semantic analysis. *Knowledge-Based Systems*. 120, (2017), 118–132.

Incorporating Prior Practice Difficulty into Performance Factors Analysis to Model Mandarin Tone Learning

Meng Cao
University of Memphis
400 Innovation Drive, Memphis,
Tennessee 38152, USA
mcao@memphis.edu

Philip I. Pavlik Jr.
University of Memphis
400 Innovation Drive, Memphis,
Tennessee 38152, USA
ppavlik@memphis.edu

Gavin M. Bidelman
University of Memphis
807 Jefferson Avenue, Memphis,
Tennessee 38105, USA
g.bidelman@memphis.edu

ABSTRACT

Determining how to select items for practice is an important task for any adaptive training system. Prior work has tried to create systems for second language learners (L2) to learn Mandarin tones, but such work does not often have a well-determined algorithm to choose items for practice. In order to develop a model for adaptive practice selection, we designed an experiment and asked L2 learners on Amazon Turk to finish a series of tone learning trials. Using this data we ranked the difficulty level of the stimuli and incorporated a quadratic function of difficulty for prior successes and failures into the Performance Factors Analysis (PFA) model. Results of logistic regression were used to compute the optimal difficulty level for each tone. For the four Mandarin tones, the optimal difficulty scores were 0.86, 0.75, 0.54 and 0.60, respectively. Crossvalidated results showed that the new PFA-Difficulty model had better performance than the original PFA model. While the advantage was not large, the new model allows for clear inferences about optimal item difficulty that can be used by an adaptive training system to select items for practice.

Keywords

Mandarin tone, Difficulty level, Performance factors analysis, Adaptive training.

1. INTRODUCTION

In Mandarin, there are four lexical tones: Tone 1, a high-level tone; Tone 2, rising tone; Tone 3, a low falling or low falling-rising tone, and Tone 4, a falling tone. It is a challenging task for second language learners (L2) to distinguish four tones [1,2]. Adaptive training system is thought to be an effective tool to train learners to acquire new skills by selecting items and adjusting presentation orders to address learner variability.

However, only a few studies have tried adaptive training systems for tone learning. Shih et al. [1] roughly selected items based on the talker-to-listener distance to control the task difficulty as they thought that the speech projected over a greater distance is easier while nearer is more difficult. Feiya Li et al. [2] used a hybrid training system which combined adaptive training with high variability phonetic training to train Japanese learners on Mandarin

tones. However, they did not provide details of how they selected the items or developed the system.

How to select materials according to learners' capability is a crucial issue. The overall goal of this study was to develop a model for adaptive practice selection for Mandarin tone learning. While prior work by Pavlik and Anderson has computed optimal practice levels [3], this work relied upon a model of spaced practice to justify the inverted U-shaped optimal difficult curve that was derived. This seems infeasible in the domain of tone learning since spacing will always be rather narrow on average with only 4 tones. Another option is to select items based on their difficulty level. Research has shown that adapting the difficulty of training based on learners' performance can improve the efficiency of training [4]. Similarly, Kelley [5] suggested that keeping an appropriate level of difficulty can make training effective. If a task is too easy, learners may learn very little probably because low levels of mental effort don't cause as much learning. While if the task is too difficult, learners may be unable to encode the experience and show weaker learning. Thus, the relationship between difficulty and learning is likely similar to the Yerkes-Dodson law (often expressed as an inverted-U function) which suggests that there is an optimal level of arousal for an optimal performance, over or under arousal reduces task performance [6]. We hypothesized that students would benefit more from example practice in a medium range of difficulty when learning tones, the relationship between difficulty and learning is also in an inverted-U shape.

Performance Factors Analysis (PFA) is a student model that traces predictions of individual students on knowledge components (KCs) using successes and failures [7], which might be helpful for developing effective adaptive technologies. KCs are "acquired units of cognitive function or structure that can be inferred from performance on a set of related tasks" [8]. Each tone can be considered as a KC. The original Performance Factors Analysis model is as follows.

$$m(i, j \in KCs, s, f) = \sum_{j \in KCs} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}) \quad (1)$$

$$p(m) = \frac{1}{1 + e^{-m}} \quad (2)$$

In Equation 1, m is a logit value representing the accumulated learning of student i on one or more KCs j . β represents the easiness of each KC. s and f track the prior successes and failures of the student on each KC. γ and ρ scale the effect of the observation counts. Equation 2 shows the standard conversion from logit to probability prediction.

Even though prior successes and failures are strong indicators of student learning, tracking the counts of prior successes and failures

Meng Cao, Philip Pavlik and Gavin Bidelman "Incorporating Prior Practice Difficulty into Performance Factors Analysis to Model Mandarin Tone Learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 516 - 519

is meaningless for item selection without some rule for what to do with the predictions once they are computed. From the results of PFA model, we can know whether correct or incorrect responses lead to more learning on each KC, but this can't be used to select items since if the success coefficient is higher than the failure coefficient, this implies us to minimize the difficulty and choose practices that maximize immediate success. However, the pedagogical theories we reviewed suggest that it is misguided. Thus, we need to revise the PFA model to capture a curvilinear effect of prior practice difficulty of learning.

To achieve this goal reliable data is needed on the learning functions for 4 tones. We designed an experiment to gather data and planned to use this data to analyze learning as a function of these prior difficulty factors.

2. METHOD

2.1 Participants

325 participants on Amazon Mechanical Turk (MTurk) finished the practice. Participants needed to meet the following criteria: be over the age of 18, have little knowledge of Mandarin tones and no hearing problems, and have successfully completed 100 MTurk tasks previously with 95% acceptance. 70 of them were excluded because of technical problems. 49 of them were excluded as their scores were lower than 0.31, which would mean a 95% likelihood of chance performance. 206 participants were used in the analysis below (Female = 90, Male = 116).

2.2 Stimuli

We synthesized varied stimuli with 4 tones, 3 syllables (ma, mo, and ya), 3 durations (400ms, 800ms, and 1200ms), 3 expansions (1.0, 1.4, and 1.8), and 2 gender voices (male and female). There were 216 stimuli. The fundamental frequency (f_0) were controlled in MATLAB and overlaid onto syllables in PRAAT [9].

2.3 Procedure

We used the Mobile Fact and Concept Training System (MoFaCTS) which is a flashcard (one-step drill) learning system that can be used to schedule experimental practice [10]. Participants needed to finish 216 trials with feedback. For each trial, they needed to listen to the tone and choose the correct answer. If the answer was correct, the system would show the feedback and then immediately went on to the next trial. However, if the answer was incorrect, the system would replay the tone and showed the correct answer for 6 seconds.

Table 1. Learning conditions

Factor	Conditions	Trials 1-72	Trials 73-216
Syllable	6	e.g., ma	e.g., ma, ya
Duration	6	e.g., 400ms	e.g., 400ms, 800ms
Expansion	6	e.g., 1.0	e.g., 1.0, 1.4
Gender	6	e.g., male	e.g., male, female

The dependent variable was learners' correctness on the trials. The experiment was a mixed between and within subject design. Tone was a within-subject variable while other variables were mixed between and within subject variables. Participants were randomly assigned to one of the 24 learning conditions (see Table 1). On each factor, there were 6 conditions counterbalancing the stimuli participants learned. The factors listed in the table did not vary

while other factors varied. For example, in a condition on syllable, participants first learned 72 trials of *ma* (3 duration \times 3 expansion \times 2 genders \times 4 tones = 72), then they learned another 144 trials which randomly mixed the first 72 trials of *ma* and another 72 trials of *ya*. The goal of this design was to look for both learning from repetition of items and transfer to similar items.

2.4 Statistical Analysis

To create a model to determine the optimal practice difficulty, we incorporated prior practice difficulty into the PFA model to see how the difficulty influences learning gains for successes and failures. Since we didn't have a quantitative theory on the exact shape of the inverted U relating learning and difficulty, we used the simple quadratic equation below because it was computationally tractable compared to alternatives. In Equation 3, y represents the effects of learning from each item, where x represents the difficulty level of the item.

$$y = ax^2 + bx \quad (3)$$

Then the new formula of Difficulty PFA model takes the following form (see Equation 4). We track the effects of prior difficulty level using quadratic equations for both prior successes and failures to replace the counts.

$$m(i, j \in KCs, s, f) = \sum_{j \in KCs} (\beta_j + \gamma_2 x_{s,i,j}^2 + \gamma_1 x_{s,i,j} + \rho_2 x_{f,i,j}^2 + \rho_1 x_{f,i,j}) \quad (4)$$

We have emphasized that keeping an optimal level of difficulty of items might be beneficial for learning. The new model can help us find at what difficulty level, learners' performance improves the most. To use the new model, we first did logistic regression to determine the difficulty of the items (Model 1 below). Then we formed the new PFA model (Model 2 below) which incorporated the prior practice difficulty and compared it with the original PFA model. Following this, we used the coefficients to graph of the optimal difficulty level for each tone. All analyses were completed in R, with source code available from the corresponding author.

3. RESULTS

Table 2. Results of logistic regression

Model	Predictors	Mean test fold R ²	Mean test fold RMSE
Model 1	Tone:Duration:Gender:Syllable:Expansion	0.1148	0.4604
Model 2	Tone:Duration+Diffeffectcor1:Tone+Diffeffectincor1:Tone+Diffeffectcor2:Tone+Diffeffectincor2:Tone	0.1872	0.4388
Model 3	Tone:Duration+CF..tonesubcor.:Tone+CF..tonesubincor.:Tone	0.1841	0.4397

Model 1 (see Table 2, where the R code : indicates interactions) was done to get a constant coefficient for all the items to represent their difficulty score. The higher the coefficient, the easier the item. Model 2 was the new model we created which incorporated prior practice difficulty into the Performance Factors Analysis model. Tone by duration interaction was used as the intercept to represent the prior easiness of the KCs. *Diffeffectcor1* is the sum of difficulty

scores of prior successes. *Diffeffectcor2* is the sum of squared difficulty levels of prior successes. Similarly, *Diffeffectincor1* and *Diffeffectincor2* represent prior failures. Their interaction in the model with tone means tracking of each of the 4 tones as individual KCs. Model 3 was constructed according to the original PFA model. *CF.tonesubcor.* and *CF.tonesubincor.* tracked the counts of prior successes and failures on each tone. In order to assess whether the results of the models could generalize to an independent data set or not, we did 10-fold cross validation 100 times for each model and calculated the mean of R^2 and RMSE (see Table 2). The crossvalidated test fold R^2 of Model 2 was larger than Model 3, and the crossvalidated test fold RMSE was smaller than Model 3 which all indicated that Model 2 had better performance than Model 3.

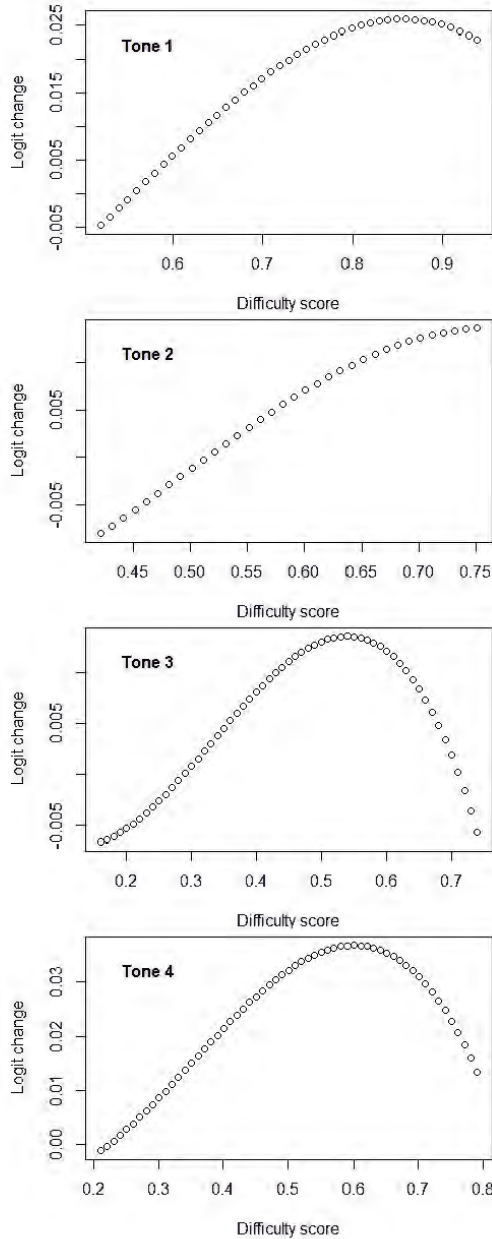


Figure 1. Plots of logit change given the difficulty level for each tone. The x-axis represents the difficulty score, and the y-axis represents the logit change. Given the difficulty value x , we can get y amount of change in logit per trial.

Our next step was to use Model 2 to infer the optimal difficulty level of items for learning. Since logistic regression tracks an underlying linearly increasing strength, “the logit”, we decided to use this as the quantity we wanted to be maximal for each repetition of practice. While more complex questions can be asked, such as considering the gain/cost, for this initial work we wanted to show only the gain function as a proof of concept that this method could be used to create pedagogically significant models.

From the results of Model 2, we used the coefficients of each parameter and computed the logit function increase or decrease as a function of difficulty on each tone (see Table 3). Then we plotted the functions and found the maximum values of logit change (see Figure 1) for the range of difficulty in the data. For Tone 1, the change in logit per trial is the largest when the difficulty score is 0.86, For Tone 2, Tone 3, and Tone 4, the difficulty scores are 0.75, 0.54 and 0.60, respectively.

Table 3. The function logit change of difficulty level

Tones	Range of difficulty (x)	The function logit change
Tone 1	0.519 to 0.940	$x*(0.276*x-0.26*(x)^2) + (1-x)*(-0.27*x+0.19*(x)^2)$
Tone 2	0.421 to 0.756	$x*(0.19*x-0.20*(x)^2) + (1-x)*(-0.19*x+0.19*(x)^2)$
Tone 3	0.160 to 0.742	$x*(0.45*x-0.57*(x)^2) + (1-x)*(-0.12*x+0.014*(x)^2)$
Tone 4	0.211 to 0.800	$x*(0.59*x-0.68*(x)^2) + (1-x)*(-0.13*x+0.017*(x)^2)$

4. DISCUSSION

In this study, we collected learners’ Mandarin tone learning data on Amazon Turk and found that tone and duration were the most important factors predicting learners’ performance, but many different factors determined difficulty for items of each tone. Then we replaced counts of prior successes and failures with the quadratic functions of the difficulty level of prior practice. Crossvalidated results showed that the performance of the revised “PFA-Difficulty” model was better than the original PFA model. Even though the quantitative benefit was not large, it is very meaningful for later item selection. Different tones have different optimal difficulty levels. For the four tones, the optimal difficulty scores were 0.86, 0.75, 0.54 and 0.60, respectively.

While we show the computation to produce these curves from the model coefficients for each tone, it may not be obvious how the underlying correctness and incorrectness each show interpretable curves as a function of difficulty. Figure 2 shows the correctness and incorrectness curves separately for tone 1 (plotted using either the 2 correctness coefficients, top, and the 2 incorrectness coefficients, bottom). As we can see, only the correctness curve (top) has an inverted U-shape individually. This follows the “Goldilocks principle” and the zone of proximal development theory which all propose that practices should be neither too easy nor too hard [11,12]. The result also makes intuitive sense, since it is commonly understood that practice needs to match a student’s current level of proficiency.

The incorrectness curve in Figure 2 (bottom) is less intuitive, but it helps to remember that PFA’s adaptive nature means that in some cases, and for our data here, the coefficient for failure is negative. The figure means that when a learner gets it wrong, the penalty is monotonically more negative as difficulty is less. This again makes

sense since we might expect that if a student gets the easiest items for a KC wrong, the proficiency estimate for that KC should be adjusted more negatively.

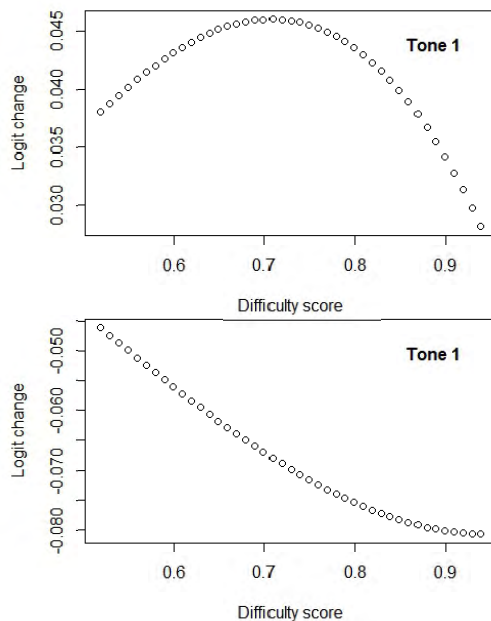


Figure 2. Correctness curve (top) and incorrectness curve (bottom) for Tone 1. The x-axis is the difficulty score, and the y-axis is the logit change.

The PFA-Difficulty model considered the difficulty level of items instead of simply counting. The quadratic function means that the relationship between practice difficulty and learning is nonlinear [5]. According to this perspective, there should be a point between maximum and minimum difficulty where learning is maximal. This may be true for multiple reasons, for example, keeping item difficulty at an intermediate level could help to maintain learners' motivation and persistence at a higher level [13].

Our finding in this study is the first step to construct an "optimal" adaptive training system. To achieve adaptive training we need to constantly adjust the difficulty level of items selected in order to keep performance near the pre-defined level [13]. However, the analysis we showed used static difficulty predictions, which means that the curves we computed are not sensitive to changes in performance. In other words, they answer the question of which specific item is optimal overall, but they do not say how the choice of the optimal item also depends on the change of performance due to learning. Intuitively, and in preliminary tests, it seems that we need to input learners' learning rate into the initial model of difficulty, so the curves will predict optimal learning as a function of performance difficulty (which includes both item difficulty and learning) instead of only item difficulty. That is the direction for current work which is out of the scope of this preliminary report.

5. CONCLUSION

This paper incorporated the prior practice difficulty into the PFA model using a quadratic function of the difficulty level and proved that the new model was a little better than the original PFA model. The new model could be considered an update or a variant of the

PFA model which could better track students' learning and provide corresponding guidance. It is particularly useful for item selection when constructing an adaptive training system since it leads to specific predictions about the most optimal item to practice, given a set of possible items for some KC.

6. ACKNOWLEDGMENTS

The present research was funded by the LearnSphere NSF grant #1443068.

7. REFERENCES

- [1] Shih, C., Lu, H.Y.D., Sun, L., Huang, J.T. and Packard, J., 2010. An adaptive training program for tone acquisition. In *Speech Prosody 2010-Fifth International Conference*.
- [2] Li, F., Xie, Y., Yu, X. and Zhang, J., 2016, October. A study on perceptual training of Japanese CSL learners to discriminate Mandarin lexical tones. In *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)* (pp. 1-5). IEEE.
- [3] Pavlik Jr., P.I. and Anderson, J.R., 2008. Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied* 14, 2, 101–117.
- [4] Landsberg, C.R., Astwood Jr, R.S., Van Buskirk, W.L., Townsend, L.N., Steinhauer, N.B. and Mercado, A.D., 2012. Review of adaptive training system techniques. *Military Psychology*, 24(2), pp.96-113.
- [5] Kelley, C.R., 1969. What is adaptive training?. *Human Factors*, 11(6), pp.547-556.
- [6] Cohen, R.A., 2011. Yerkes–Dodson Law. *Encyclopedia of clinical neuropsychology*, pp.2737-2738.
- [7] Pavlik Jr, P.I., Cen, H. and Koedinger, K.R., 2009. Performance Factors Analysis--A New Alternative to Knowledge Tracing. *Online Submission*.
- [8] K. R. Koedinger, A. T. Corbett, and C. Perfetti, "The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning," *Cognitive science*, vol. 36, no. 5, pp. 757–798, 2012.
- [9] G. M. Bidelman and C.-C. Lee, "Effects of language experience and stimulus context on the neural organization and categorical perception of speech," *NeuroImage*, vol. 120, pp. 191 – 200, Oct. 2015.
- [10] Pavlik, P.I., Kelly, C. and Maass, J.K., 2016, June. The mobile fact and concept training system (MoFaCTS). In *International Conference on Intelligent Tutoring Systems* (pp. 247-253). Springer, Cham.
- [11] Halpern, D.F., Graesser, A. and Hakel, M., 25. Learning principles to guide pedagogy and the design of learning environments. *Washington, DC: Association of Psychological Science Taskforce on Lifelong Learning at Work and at Home*.
- [12] Vygotsky, L., 1986. *Thought and Language*. MIT Press, Cambridge, Mass.
- [13] Treleaven, A.J., 2016. *Improving reading performance in peripheral vision: An adaptive training method* (Doctoral dissertation, The Ohio State University).

Parent as a Companion for Solving Challenging Math Problems: Insights from Multi-modal Observational Data

Lujie Chen
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA USA
lujiec@andrew.cmu.edu

Eva Gjekmarkaj
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA USA
egjergji@andrew.cmu.edu

Artur Dubrawski
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA USA
awd@cs.cmu.edu

ABSTRACT

Parents play key roles in forming their children's trajectories of development. Parental involvement has been studied in early grade reading and math facts' practices. However, their hands-on engagement with children's math problem solving activities, in particular with the goal to develop higher order problem solving skills and promote perseverance behaviors, has not received substantial attention yet. With the goal to understand the dynamic interplay between children's affective and cognitive processes and parents' support, we have collected multi-modal data of multiple parent-child dyads where parents serve as companions for their children while working through challenging math problems at home. In this paper, we report some initial findings with this data which has been annotated with a combination of automatic and manual methods.

Keywords

math problem solving, multi-modal learning analytics

1. INTRODUCTION

Different from math practices, non-routine math problems are those for which no immediate solutions are available. Whether or not one may eventually solve such a problem depends not only on the activation of prior conceptual and procedural knowledge, but also on the mastery of general problem solving strategies, meta-cognitive skills, as well as the habit of mind to persevere through impasses or uncertainties.

Though researchers have explored in-school interventions for supporting development of perseverance [3], and the common core standard lists mathematical perseverance as one of the math practice standards since its inception, it comes however without a clear guideline on how to operationalize it in a real-world school environment with constrained resources.

Lujie Chen, Eva Gjekmarkaj and Artur Dubrawski "Parent as a Companion for Solving Challenging Math Problems: Insights from Multi-modal Observational Data" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 520 - 523

One alternative answer to that challenge is to look for solutions beyond schools. Parents are typically not trained as professional educators, however, from daily interactions they often gain rich insights into their own child's stable traits such as personality, interest, self-efficacy and self-control, which provide valuable information in estimating the child's tolerance to frustration - an important parameter to guide the support.

The ultimate goal of our project is to explore a model of at-home intervention where parents are motivated to engage with their own child in solving challenging math problems on a regular basis. Perseverance habits, though believed to be malleable, take time and consistency to develop. The main challenge in implementing effective interventions is thus the consistency of time investment given the busy schedules of modern parents. We envision a Fitbit¹-like tracking tool that can measure and report a child's exposure to struggle, that can analyze and give feedback on parents' coaching strategies, and that can motivate and engage both the parent and the child by providing data driven analysis to stimulate ongoing conversation, reflection, and improvement.

As the first step toward this vision, we set out to explore the feasibility of the envisioned work flow by collecting and analyzing high resolution observational data of parent-child dyads at home while they are working on challenging math problems together. From those baseline data, we aim to develop a suite of relevant metrics and analytical methods that can facilitate the reflection and improvement for the child and parent. In addition, we explore the opportunities and practical challenges for automating the analysis by leveraging machine learning techniques. This paper reports the first attempt along these lines of investigation.

2. RELATED WORK

Tutorial dialogues with expert tutors have been studied extensively to identify effective tutoring strategies with more recent work contextualizing the students' emotional profiles [4]. In addition, work has been done to study negative emotions such as confusion and frustration when students are interacting with computer systems [1]. Our work builds upon those studies, however with a different goal: to model parents as imperfect tutors and help them to become better, along with their pursuit of promoting perseverance behavior.

¹<https://www.fitbit.com/>

ior in their children via solving challenging math problems at home. Our work also builds on rich mathematical problem solving literature. Though those studies mostly focus on older student population (high school and beyond), their frameworks of math problem solving are still relevant.

3. STUDY PROTOCOL

Children within age ranges of eight to twelve years old (or third to sixth grades) and their parents are recruited from a local community. They meet at least one of the two criteria: (1) The child shows interest in math problem solving; (2) The parent is interested in developing math problem solving skills in his or her own child. The data were collected at home by parents using a webcam (Logitech 1080P) and digital pen (Livescribe 2GB Echo Smartpen).

In each session, the child works through one problem and the parent provides support standing by on the child's side as needed. Parents are advised to choose the problems that are most likely to challenge their children thus inducing a certain amount of confusion or frustration.

4. DATA SET OVERVIEW

Data from 42 sessions were collected, however, data from 6 sessions were excluded due to child's limited amount of exposure to struggles. For the remaining 36 videos, the cumulative duration of is about 307 minutes, with a mean duration of 7.9 minutes per session. The shortest session lasts about 2.7 minutes while the longest one is about 21 minutes.

We used OpenFace [2] tool to extract features related to the child's head movements and rotations (i.e. roll, pitch and yaw), and eye gaze orientations. In addition, we also used the FACET tool to extract estimation of 6 basic emotions as well as confusion and frustration.

We annotate talking episodes for both child (child-talk) and parent (parent-talk). The annotation is done at the utterance level. In addition, we annotate the child's eye gazes which are round trips of child's head poses and eye gaze changes toward his or her parent and back. Based on those annotations of discrete events and intervals, we compute continuous intensity measures derived from event intervals smoothed by Gaussian kernels over time. This measure combines event frequency and duration, so that many short episodes of talking are equivalent to a few long episodes.

4.1 An Example Session

Figure 1 shows one example session with a subset of features (affect estimations for joy, confusion and frustration, writing speed and eye blinks), together with annotation for child and parent's talk and child's eye gazes (presented as intensity measures mentioned above). This session lasts for about 8 minutes. The child was working on a word problem that requires him to piece together the information from the problem and reason through it in order to get the answer. His mother provided moderate amount of support along the way. Details of the progression are as follows:

- Phase A: The child reads the problem out loud during the first 17 seconds;

- Phase B: The child jots down some notes on paper (as seen by spikes in writing speed) and tries to make sense of the information given, while brief moments of confusion and frustration are noted. This episode lasts for about 1 minute;
- Phase C: He continues to reason through the problem, noted by the decrease of eye blinks during this period of time, which might be linked to an increase in cognitive load. This episode lasts about 1.5 minutes;
- Phase D: This phase starts with the parent's suggestion to re-read the problem which results in a moderate amount of intervention where the parent guides the child through the reasoning process and provides confirmation from time to time. The child's eye gaze occurs concurrently with the interaction. It is worthy to note that the child exhibits a substantial amount of persistent confusion and a few brief episodes of frustration. The eye blinks in the first half of the phase are low suggesting high intensity of thinking. This phase ends at the 7th minute where the child completed solving the problem and wrote down the solution.
- Phase E: This is a reflection phase in which the parent suggests an alternative way in solving the problem using a number line. The child listens attentively as manifested by the high intensity of concurrent eye gaze episodes.

It is worth mentioning that the two spikes of joy during phases C and D are indeed "false positives". The child was in a neutral emotional state while speaking sentences with phrases such as "adding eight" which was mistakenly picked up by the affect detector as smiles, possibly triggered by the activation of two joy/happiness related facial action units "cheek raiser"(AU6) and "lip corner puller" (AU12).

5. RESULTS

In this section, we present preliminary analysis of the data set, mainly based on the manual annotations of the dialogue interactions of the parent-child pairs, as well as each individual child's eye gaze events. Where necessary, we incorporate automatically derived features such as the emotional states.

5.1 Parent-child verbal interaction patterns

Figure 2 shows the ratio between the amount of the child's talk and parent's talk summarized at session levels, grouped by subjects. This ratio is a proxy for the amount of parents' intervention during the problem solving process. For example, a session where the child is driving the process is expected to have a higher ratio whereby for sessions in which parents take control are likely to have lower ratios. This is the case for subjects A002 and A003, who are siblings coached by the same parent that tends to spend a large chunk of time working through problems with the child at each detailed step. On the contrary, the parent for subject A004 seats back and allows her child "struggle" first and only intervenes as necessary. When intervening, she often gave high level guidance and brief feedback along the way. Also of note, there are between-session variations for the same subject as the case for subject A005. This could possibly be explained by the varying challenge levels of the problems. If

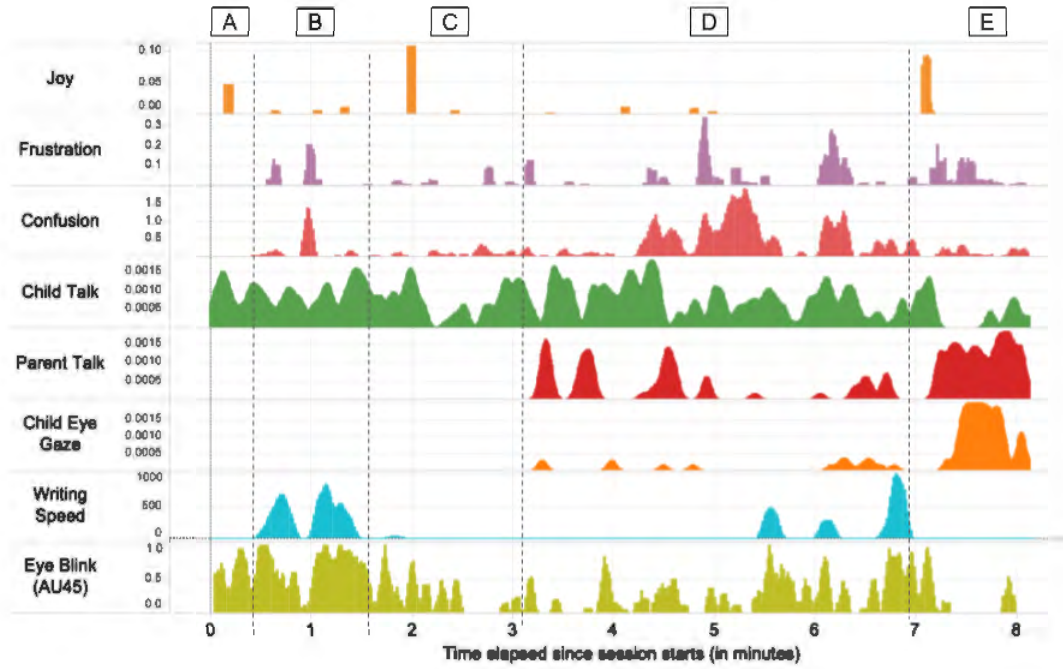


Figure 1: An example session illustrating the evolution of emotional profiles (Joy, Frustration and Confusion), parent and child’s talking episodes, event intensity, writing speed and eye blinks estimation, within the context of problem solving phases from A to E.

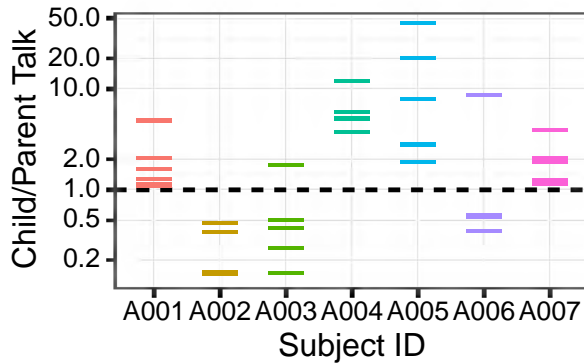


Figure 2: The ratio between amount of child-talk and parent-talk per session, grouped by subjects. The vertical axis is logarithmically scaled.

the problem is relatively easy, the child does not need much help and the ratio is high, otherwise, parents need to offer more hands-on help which drives the ratio down.

5.2 Child Gaze Analysis

Gazes are natural ingredients of inter-personal communications which may carry rich meaning and trigger a variety of responses. During the problem solving process, the child may use gazes to express confidence, confusion or frustration, or signal the need for support. Here, we use 524 instances of individual child’s gazes toward their parents as “anchoring points” to infer their cognitive and affective states, as well as the parents’ response patterns. Since gazes

are closely related to the verbal interaction patterns between the parent and the child, we differentiate gazes that occur when the child is the driver of the conversation (“talking gazes”) and those that occur when the parent is talking and the child is listening (“listening gazes”). However, if we note that a child looks at his or her parent without any accompanying talking episodes, and possibly is waiting for an approval or a confirmation response from the parent, we then label those as “approval gazes”.

5.2.1 Gaze Type Distribution

Figure 3 summarizes the gaze-type distribution by subject. It is not surprising to see that talking gaze accounts for the majority of most subjects except for subjects A002 and A003, which could be attributed to the fact that their parent was in the “drivers’ seat” for the majority of time, as revealed by the child-parent talk ratios in Figure 2. As noted, “approval gazes” are relatively rare, and they may be completely missing for some subjects such as A002, A004 and A005.

5.2.2 Analysis of Talking Gazes

For 295 talking gazes, we further annotate whether the child’s utterances are displaying reasoning which are suggestive of deep thinking processes or they are simply reading out loud the problems or describing the procedures such as solving an equation. Furthermore, for those talking gazes to which parents responded within 5 seconds ($n=203$), we annotate whether or not the feedback that the parents gave indicated positive confirmations. Examples of positive feedback are those with key words such as “okay”, “good”, “yes”, “you are right” etc.

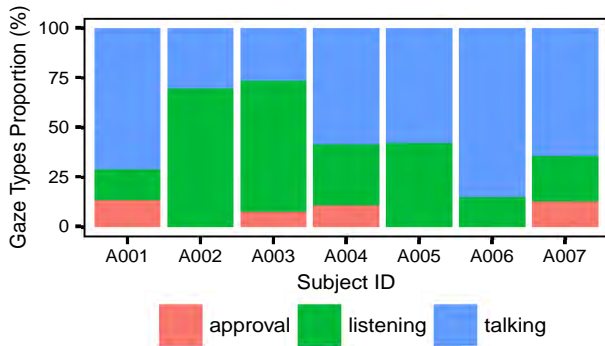


Figure 3: Relative frequencies of gaze types, grouped by subjects.

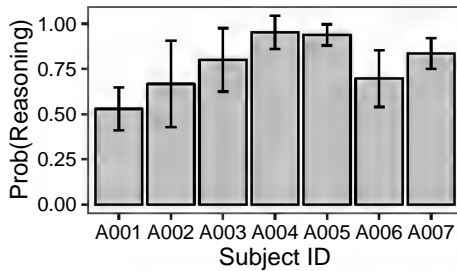


Figure 4: Proportion of talking gazes where child displays reasoning.

5.2.3 Whether a Child is Displaying Reasoning?

Figure 4 shows the likelihood that a child is displaying reasoning given a talking gaze, aggregated for each subject. As shown, for subjects A004 and A005, they are more likely than others to display reasoning. Interestingly, as noted in Figure 2, those two subjects are the ones with above-the-average child-to-parent talk ratio. The connection is not clear however, because these subjects also exhibit a more mature problem solving personalities, and are thus conduct more independent thinking, all the while their parents' decision to stay back forces them to think more on their own.

5.2.4 Whether Parent Gives Positive Feedback?

In Figure 5 we summarize the likelihood of a parent giving a positive response to the child's talking gaze, grouped by

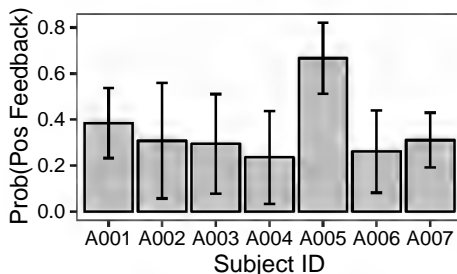


Figure 5: The likelihood of the parent giving positive feedback when they respond.

subject. We note that the parent for subject A005 tends to give more positive feedback than others. It may suggest that the child is progressing nicely or that the parent chose to be more encouraging than average. Regardless of the reason, positive reinforcement like this is likely to have a positive effect on the child's experience, encouraging them to take on more challenging problems in the future.

6. DISCUSSION AND FUTURE WORK

In this paper, we reported data collection and preliminary analyses of 36 sessions of math problem solving by primary school children obtained from 7 parent-child dyads recorded at home by parents. We found that even with simple metrics derived from child and parent talking patterns and eye gazes, we may probe insights into a child's problem solving process when supported by a parent. Specifically, we were able to obtain session-level estimates of the amount of parents' support being offered by looking at the talking patterns. We were also able to get a deep understanding of a child's overall cognitive processes and emotional experience from a set of analyses anchored on the child's gazes. Future work will take those analyses to the level of moment-by-moment, so that we could have a better understanding of the dynamic interplay between parents' real time decisions in terms of the timing and the types of support, and the child's responses and experiences. Ultimately, the research presented in this paper will help develop real-time assistance tools for both the parent and the child, in order to guide both towards more effective home practice experiences.

7. ACKNOWLEDGEMENT

The research reported here was supported, in whole or in part, by the Institute of Education Sciences, U.S. Department of Education, through grant R305B150008 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent the views of the Institute or the U.S. Department of Education.

8. REFERENCES

- [1] R. S. Baker, S. K. D'Mello, M. M. T. Rodrigo, and A. C. Graesser. Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive-affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies*, 68(4):223–241, 2010.
- [2] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L.-P. Morency. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 59–66. IEEE, 2018.
- [3] H. Bass and D. L. Ball. Beyond "you can do it!": Developing mathematical perseverance in elementary school. *The Collected Papers*. Chicago, IL: Spencer Foundation. Article Available Online [<http://www.spencer.org/collected-papers-april-2015>], 2015.
- [4] B. Lehman, M. Matthews, S. D'Mello, and N. Person. What are you feeling? investigating student affective states during expert human tutoring sessions. In *International conference on intelligent tutoring systems*, pages 50–59. Springer, 2008.

Gender Differences in Work-Integrated Learning Assessments

Shivangi Chopra, Abeer Khan, Melicaalsadat Mirsafian, and Lukasz Golab
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
{s9chopra,a383khan,mmirsafi,lgolab}@uwaterloo.ca

ABSTRACT

We analyze gender differences in performance assessments and student satisfaction data for nearly 9,000 students enrolled in undergraduate work-integrated learning (WIL) programs in a large North American university. Our analysis leads to two main findings. First, women receive slightly higher performance appraisals from their WIL employers. Second, men appear to be more satisfied with their WIL experiences, especially with compensation, networking opportunities and the ability to make meaningful contributions, while women appear to be happier with the availability of employer support.

1. INTRODUCTION

The gender gap in Science, Technology, Engineering and Mathematics (STEM) is well-documented: studies have shown that fewer women obtain STEM degrees and continue with STEM careers. Some researchers found that work experiences drive attrition more than other factors [5, 7]. However, as noted by Kauhanen et al., [8] while research on gender differences focuses on later career stages, *early* career experiences can greatly affect subsequent career choices. To fill this gap, we investigate gender differences in early engineering careers. We observe that work-integrated learning (WIL), or co-operative (co-op) education, has become part of undergraduate engineering curricula worldwide. In WIL programs, students alternate between classroom study terms and work terms. These work terms correspond to students' first experiences in the engineering workplace.

Our analysis is enabled by access to unique datasets, covering a year of WIL performance appraisals and student satisfaction data for nearly 9,000 undergraduate engineering students in a large university. Our main findings are as follows. In terms of perceived competencies, women tend to receive higher performance appraisals from their WIL employers. In terms of satisfaction, a focused analysis on a second batch of students suggests that men appear to be more satisfied with their WIL experiences, especially with compensation, networking opportunities and the ability to make meaningful contributions, while women appear to be happier with the availability of employer support. However, these results

Shivangi Chopra, Abeer Khan, Melicaalsadat Mirsafian and Lukasz Golab "Gender Differences in Work-Integrated Learning Assessments" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 524 - 527

Table 1: Student performance evaluation criteria

1. Interest in Work
2. Ability to Learn
3. Quality of Work
4. Quantity of Work
5. Problem Solving
6. Teamwork
7. Dependability
8. Response to Supervision
9. Reflection
10. Resourcefulness
11. Ethical Behaviour
12. Appreciation of Diversity
13. Entrepreneurial Orientation
14. Written Communication
15. Oral Communication
16. Interpersonal Communication

should be interpreted with caution: they are based on one year of WIL data from a North American institution, but gender differences are also influenced by cultural factors.

2. DATA AND METHODS

We analyze one year of WIL performance appraisal and student satisfaction data (September 2015 to August 2016) for 8,956 students enrolled in engineering or computer science programs (abbreviated as ENG). At the end of each work term, WIL employers give their student employees an overall evaluation on a 7 point scale (Unsatisfactory, Marginal, Satisfactory, Good, Very Good, Excellent and Outstanding Performance), and a detailed evaluation on 16 criteria listed in Table 1, also on a 7 point scale, grouped into Developing (1-2), Good (3-5) and Superior (6-7), or "N/A" indicating not applicable. The evaluator's gender is unknown. Additionally, students rate their employers from one to ten (ten being most satisfied).

The academic programs within ENG are Computer Science and Engineering (38% of ENG), and Mechanical (21%), Industrial (9%), Electrical (8%), Chemical (8%), Civil (7%), Environment (5%), Nanotechnology (5%) and Biomedical (1%) Engineering. We consider three groups of students: all ENG students, only Computer

Table 2: Gender breakdown by program

Group	Seniority	Students	% Men	% Women
ENG	All	8956	77%	23%
	Junior	3828	74%	26%
	Senior	2144	81%	19%
COMP	All	3381	84%	16%
	Junior	1523	82%	18%
	Senior	693	87%	13%
MECH	All	1843	87%	13%
	Junior	780	83%	17%
	Senior	490	90%	10%

Table 3: Student satisfaction criteria

1. Availability of employer support
2. Opportunities to learn or develop new skills
3. Opportunities to make meaningful contributions at work
4. Opportunities to expand your professional network
5. Appropriate compensation and/or benefits
6. How closely was the work related to your academic program
7. How closely was the work related to the skills you are developing at university

Science and Engineering students (abbreviated COMP) and only Mechanical Engineering students (abbreviated MECH). We single out COMP and MECH since these are the two largest programs in the dataset. Since we are analyzing students' work experiences, we measure student seniority in terms of the number of work terms completed rather than the academic level: junior students are those who have completed 0 or 1 work terms and senior students are those who have completed at least 4 work terms. The sizes and the gender mix of the different populations under study are summarized in Table 2.

Additionally, we analyze two semesters of data (4,888 students) from a pilot program run from January to August 2017 to explore students' satisfaction with their work terms. In addition to giving an overall satisfaction score, students provided a score from 1 to 5 (with 5 being most satisfied) for the seven questions listed in Table 3. This dataset overlaps with the other 2015/2016 dataset, but some students from the other dataset have graduated by 2017, and there are new students who enrolled in Fall 2016 and had their first work terms in 2017.

We start by comparing the average overall evaluations of men and women by their WIL employers using the Mann-Whitney test. Then, for each of the 16 evaluation criteria listed in Table 1:

1. We compare the average scores of men and women using the Mann-Whitney test,
2. We use a proportion test to compare the fraction of men and women receiving "Developing", "Good" and "Superior" scores,
3. We use a proportion test to compare the fraction of men and women receiving "N/A".

We chose the Mann-Whitney test because it is suitable for the Likert scale used in performance evaluations. Next, to examine differences in satisfaction from the student's perspective, we use the 2015/2016 dataset to calculate average student's evaluations of the employers and compare them using the Mann-Whitney test (again, because of the ordinal nature of the data). We use the same method on the 2017 dataset to identify significant differences in the seven specific satisfaction criteria (Table 3). We repeat each analysis on all, junior and senior students in ENG overall, COMP, and MECH. We use a P-value of 0.05 for all tests.

3. RESULTS

3.1 Workplace Performance Evaluations

Table 4 shows gender differences in all of ENG in the overall performance rating and in the 16 evaluation criteria. We report results of the Mann-Whitney test for the differences of means as well as results of the proportion tests for the fractions of students whose skills were rated as "Developing", "Good", "Superior" and "N/A" (recall Section 2). Furthermore, Table 5 shows the Mann-Whitney tests separately for COMP and MECH, and for juniors and seniors. For statistically significant differences, we report the absolute difference (for the Mann Whitney test) or the percentage difference (for proportion tests), and specify M or F to indicate whether the number was higher for men or for women; hyphens indicate no statistically significant difference. We omit the proportion test results from Table 4 as they produced similar trends as the Mann-Whitney results shown.

We start with the overall performance rating. According to Tables 4 and 5, women receive higher overall ratings in all of ENG and in MECH, but there is no significant difference in COMP, and there is no significant difference between any group of senior women and senior men.

Next, we examine the 16 evaluation criteria. Table 4 shows that in all of ENG, women are rated more highly (and more likely to be rated "Superior") than men on most criteria. Table 5 shows similar trends for both junior and senior women. On the other hand, all men and junior men are rated more highly on resourcefulness and entrepreneurial orientation, but this trend does not persist in senior men. We see no difference in ability to learn and problem solving. Furthermore, the percentage of men who received "N/A" for teamwork, ethical behavior, appreciation of diversity and interpersonal communication is significantly higher than the percentage of women.

Zooming in on COMP, Table 5 shows that all men, junior men and senior men are rated more highly than women on entrepreneurial orientation, with other criteria showing no difference (especially for junior women) or some differences in favour of women (especially for senior women). On the other hand, there are no significant differences in entrepreneurial orientation in MECH. Furthermore, similar to COMP, MECH women are rated more highly than men on several criteria, especially senior MECH women.

3.2 Satisfaction of Students

Table 6 shows significant differences in students' overall satisfaction with WIL using the 2015/2016 dataset; we use the same notation as before. Men appear to be more satisfied in all of ENG and all of COMP. Breaking down by seniority, senior men in all of ENG and in MECH give higher satisfaction scores, but other groups do not show any significant differences.

Table 4: Statistically significant differences between evaluation scores received by men and women in ENG overall

Row	Criteria	Mann Whitney Test of Mean	Proportion test of Developing (%)	Proportion test of Good (%)	Proportion test of Superior (%)	Proportion test of N/A values (%)
1	Interest in Work	F0.08	M0.49	M2.66	F3.17	-
2	Ability to Learn	-	-	-	-	-
3	Quality of Work	F0.12	M0.57	M4.39	F5.08	-
4	Quantity of Work	F0.13	M0.66	M4.9	F5.69	-
5	Problem Solving	-	M0.84	F2.59	-	-
6	Teamwork	F0.16	-	M5.8	F7.02	M0.87
7	Dependability	F0.15	-	M5.88	F6.25	-
8	Response to Supervision	F0.10	-	M3.82	F4.37	-
9	Reflection	F0.10	M0.51	M3.45	F4.45	-
10	Resourcefulness	M0.03	-	-	-	-
11	Ethical Behavior	F0.09	-	-	F5.42	M3.62
12	Appreciation of Diversity	F0.11	-	M2.68	F8.13	M5.43
13	Entrepreneurial Orientation	M0.07	M0.66	F2.46	M3.21	-
14	Written Communication	F0.17	-	M7.42	F8.21	-
15	Oral Communication	F0.09	-	M3.59	F3.59	-
16	Interpersonal Communication	F0.17	M0.44	M6.11	F6.77	M0.22
Overall Performance Rating		F0.08				

Table 5: Statistically significant differences between evaluation scores received by men and women

Row	Criteria	ENG			COMP			MECH		
		All	Junior	Senior	All	Junior	Senior	All	Junior	Senior
1	Interest in Work	F0.08	F0.09	-	-	-	-	-	-	-
2	Ability to Learn	-	-	-	-	-	-	F0.17	-	-
3	Quality of Work	F0.12	F0.14	F0.13	F0.12	-	-	F0.16	-	-
4	Quantity of Work	F0.13	F0.16	-	-	-	-	F0.17	-	-
5	Problem Solving	-	-	-	-	-	-	F0.19	-	-
6	Teamwork	F0.16	F0.14	F0.20	F0.16	-	F0.29	F0.15	-	-
7	Dependability	F0.15	F0.15	F0.16	F0.14	-	F0.21	F0.17	-	-
8	Response to Supervision	F0.10	F0.10	F0.14	F0.16	F0.12	F0.21	F0.13	-	-
9	Reflection	F0.10	-	F0.17	F0.11	-	F0.26	F0.17	-	F0.27
10	Resourcefulness	M0.03	M0.04	-	-	-	-	-	-	-
11	Ethical Behavior	F0.09	-	F0.14	F0.13	-	-	-	-	-
12	Appreciation of Diversity	F0.11	F0.10	F0.15	-	-	-	F0.16	-	F0.30
13	Entrepreneurial Orientation	M0.07	M0.09	-	M0.13	M0.16	M0.26	-	-	-
14	Written Communication	F0.17	F0.14	F0.19	F0.10	-	-	F0.23	-	F0.25
15	Oral Communication	F0.09	F0.07	-	-	-	-	-	-	-
16	Interpersonal Communication	F0.17	F0.15	F0.23	F0.12	-	F0.28	F0.25	F0.19	F0.34
Overall Performance Rating		F0.08	F0.12	-	-	-	-	F0.19	F0.27	-

Finally, Table 7 shows significant differences in students' overall satisfaction and the seven detailed satisfaction scores using the 2017 dataset. Overall satisfaction is again higher for all ENG men, but this trend does not carry over to any subgroups. COMP women (but not senior women in isolation) give higher scores on availability of employer support, with other satisfaction criteria either showing no difference or a difference in favour of men. In particular, men appear more satisfied than women with opportunities to develop their professional network and do work more closely related to their academic program. Additionally, junior and overall ENG men report more opportunities to make meaningful contributions than women. Senior COMP men's average scores for receiving appropriate compensation are 0.23 higher (on a scale of 5) than senior COMP women's, which is the highest reported statistically significant difference of means in this analysis.

4. DISCUSSION AND CONCLUSIONS

In terms of workplace evaluations (Table 4 and 5), we found that women tend to be evaluated more highly than men. One possible

explanation is that women who decide to pursue male-dominated degrees are likely to be highly qualified; e.g., one study found that more men than women with low high school mathematics scores pursue STEM degrees [6]. Specifically, we found that women tend to be evaluated more highly on written, oral, and interpersonal communication. Similarly, Wang et al. [15] found that girls are more likely to possess *both* high mathematical and verbal abilities, and boys are more likely to demonstrate higher mathematical abilities relative to their verbal abilities. We also note the higher evaluation scores women receive on teamwork. A recent report on collaborative problem solving from the Programme for International Student Assessment (PISA) similarly found that girls outperform boys in collaborative problem solving in several countries [11]. This difference suggests the need for further investigation, especially with the growing awareness of the importance of collaborative efforts, even in traditionally competitive fields such as STEM [3].

On the other hand, we found that men in computing are perceived as having a stronger entrepreneurial orientation than women. Re-

Table 6: Gender differences in overall work term satisfaction: 2015/2016 dataset

ENG	All COMP	MECH	ENG	Junior COMP	MECH	ENG	Senior COMP	MECH
M0.12	M0.10	-	-	-	-	M0.18	-	M0.57

Table 7: Gender differences in overall work term satisfaction and satisfaction with specific aspects of WIL: 2017 dataset

Question	ENG	All COMP	MECH	ENG	Junior COMP	MECH	ENG	Senior COMP	MECH
Availability of employer support	-	F0.1	-	-	F0.1	-	-	-	-
Opportunities to learn or develop new skills	-	-	-	-	-	-	-	-	-
Opportunities to make meaningful contributions	M0.09	-	-	M0.11	-	-	-	-	-
Opportunities to expand professional network	M0.06	M0.08	M0.13	-	-	M0.2	-	M0.21	-
Appropriate compensation and/or benefits	-	-	-	-	-	-	-	M0.23	-
Work related to academic program	M0.1	-	-	M0.12	-	-	M0.11	M0.14	-
Work related to skills developed at university	-	-	-	-	-	-	-	-	-
Overall Satisfaction	M0.08	-	-	-	-	-	-	-	-

lated work on risk-taking presents conflicting reports on how risk averse men and women are [10]. There is also recent work that analyzed STEM alumni and found that the university under study produced fewer female entrepreneurs [1]. Given the importance of entrepreneurship in today's economy, it is interesting to note that any group, men or women, receive higher evaluations in this area.

In Section 3.2, we discovered gender differences in students' evaluations of their WIL experiences. Men appear to be more satisfied, especially with opportunities to make meaningful contributions at work, expanding their professional network, and working on topics related to what they learned in the classroom. Additionally, senior men in computing were more likely to report receiving appropriate compensation than women. These results agree with prior, largely qualitative, work on gender differences in workplace experiences. In particular, prior work found evidence of men receiving more opportunities (including to network and contribute meaningfully to their work) and fair compensation [2, 12, 13, 14, 4].

In our analysis of students' evaluations of their employers, the only difference in favour of women was in the availability of employer support, observed mainly in junior women in computing. Assuming that "employer support" is related to "mentorship", this result does not fall in line with prior work that found women to receive less mentoring than their male peers [2, 9]. These differences are important to examine further as they may impact young engineers' career trajectories: there is evidence that dissatisfaction over pay and working conditions can explain the higher rate of attrition for women in STEM as compared to men [7].

5. REFERENCES

- [1] A. Andrade, S. Chopra, B. Nurlybayev, and L. Golab. Quantifying the impact of entrepreneurship on cooperative education job creation. *International Journal of Work-Integrated Learning*, 19(1):51–68, 2018.
- [2] C. W. Berheide, L. Christenson, R. Linden, and U. Bray. Gender differences in promotion experiences at two elite private liberal arts colleges in the United States. *Forum on Public Policy Online*, 2013(1).
- [3] M. Borrego, J. Karlin, L. D. McNair, and K. Beddoes. Team effectiveness theory from industrial and organizational psychology applied to engineering student project teams: A research review. *Journal of Engineering Education*, 102(4):472–512, 2013.
- [4] S. K. Gardner and A. Blackstone. Putting in your time: Faculty experiences in the process of promotion to professor. *Innovative Higher Education*, 38(5):411–425, 2013.
- [5] J. L. Glass, S. Sassler, Y. Levitte, and K. M. Michelmore. What's so special about STEM? a comparison of women's retention in stem and professional occupations. *Social forces*, 92(2):723–756, 2013.
- [6] D. Hango. Gender differences in science, technology, engineering, mathematics, and computer science (STEM) programs at university. *Insights on Canadian Society*, 12 2013.
- [7] J. Hunt. Why do women leave science and engineering? *ILR Review*, 69(1):199–226, 2016.
- [8] A. Kauhanen and S. Napari. Gender differences in careers. *Annals of Economics and Statistics*, (117/118):61–88, 2015.
- [9] C. A. Moss-Racusin, J. F. Dovidio, V. L. Brescoll, M. J. Graham, and J. Handelsman. Science faculty's subtle gender biases favor male students. *Proceedings of the National Academy of Sciences*, 109(41):16474–16479, 2012.
- [10] J. A. Nelson. Are women really more risk-averse than men? a re-analysis of the literature using expanded methods. *Journal of Economic Surveys*, 29(3):566–585, 2015.
- [11] OECD. Collaborative problem solving. *PISA in Focus*, No. 78, OECD Publishing, Paris, 2017.
- [12] G. Panther, K. Beddoes, and C. Llewellyn. Salary negotiations and gender in engineering education. In *2018 ASEE Annual Conference & Exposition*, 2018.
- [13] C. Seron, S. S. Silbey, E. Cech, and B. Rubineau. Persistence is cultural: Professional socialization and the reproduction of sex segregation. *Work and Occupations*, 43(2):178–214, 2016.
- [14] K. N. Smith and J. G. Gayles. Girl power: Gendered academic and workplace experiences of college women in engineering. *Social Sciences*, 7(1):11, 2018.
- [15] M.-T. Wang and J. L. Degol. Gender gap in science, technology, engineering, and mathematics (STEM): Current knowledge, implications for practice, policy, and future directions. *Educational Psychology Review*, 29(1), 2017.

Individual Differences in Student Learning Aid Usage

Andréa K. Davis

Pearson

50 California St.

San Francisco, CA 94111

andrea.davis1@pearson.com

Yun Jin Rho

Pearson

501 Boylston St.

Boston, MA 02116

yunjin.rho@pearson.com

Daniel Furr

Pearson

501 Boylston St.

Boston, MA 02116

daniel.furr@pearson.com

ABSTRACT

There are various types of learning aids available to students while solving a problem in an online learning platform, Mastering Chemistry. Among the learning aids, we investigated two that had high usage: *Hints* and *Provided Solutions*. The main research question was how student learning aid usage differed between higher and lower performing students. To answer this question, we explored clustering students by performance metrics into higher and lower performers, and then looking at each group's process as they solved a problem. This method can be used to explore individual differences at a broad level as to how students go about solving problems.

Keywords

Learning aids, individual differences

1. INTRODUCTION

Research on problem solving has provided evidence that children and adults can use a variety of strategies for solving problems.^{1,2} These strategy differences result in different paths through the problem; for example, some students may choose to attempt the problem immediately, others to first request a learning aid, or following an incorrect attempt to immediately reattempt vs. to request a learning aid before reattempting.

One influence on strategy differences may be the performance level of the student, with higher performing students taking one strategy, and lower performers following a different one. For example, prior work has shown expert children use their knowledge in a more efficient way than novice children.³

This idea forms the following exploratory questions: How do differently performing students use learning aids? Do they have different strategies when solving a problem? An online learning platform, Mastering Chemistry, offers *Hints* and *Provided Solutions* as learning aids to students to help them during formative assessments. But they can also be used to avoid doing the more difficult work of solving a problem oneself. Higher performing students might be more motivated to solve the problem themselves, a relationship which could either be characteristic of higher

performance or lead to higher performance (doing more challenging work leads to better memory encoding⁴). Do higher performing students use hint and solutions differently than lower performing students?

1.1 Prior work on learning aid timing

In Mastering Chemistry, students choose when to request learning aids. As previously mentioned, differences in when students request a learning aid could either be characteristic of higher performance, or it could lead to higher performance. Indeed, a great deal of prior work has examined the timing of feedback and its effect on learning.^{5,6,7,8,9,10} In this work, a contrast is usually made between *immediate* feedback, given immediately after a student's response, and *delayed* feedback, given minutes, hours, or longer after a student's response. There are conflicting results about which timing of feedback is more effective for learning⁵, though it seems that delayed feedback may be more effective for high achieving students while immediate feedback is more effective for low achieving students⁶. However, timing in terms of receiving feedback prior to submitting a response to a problem vs. after responding, as well as feedback in a more natural setting in which students choose when they receive feedback, has been less studied. Thus, this work also takes an initial step toward examining the relationships between individual differences in the problem-solving process, the timing of feedback before and after submitting a response, and student performance.

2. METHODS

2.1 Sample

Two semesters in 2017-2018 of a university chemistry course taught by a single instructor were chosen for this analysis. The course was chosen because all assignments and problems had the same course implementation settings, rather than varying implementation settings between assignments, so that students' incentives to use available learning aids would not vary as a result of the implementation settings. There was a total of 1896 in the sample across both semesters, with 946 students in one semester and 950 in the other semester.

2.2 Procedure

This exploratory analysis takes the following approach to answer the research questions presented in the previous sections. First, students are segmented into 2 groups, higher and lower performing students. Segmenting students was done with k-means clustering with the following variables for each student: performance (measured by percent correct on first try), persistence (measured by percent correct on all problem parts), conscientiousness (measured by percent problem parts submitted), and consistency (inverse

Andrea Davis, Yun Jin Rho and Daniel Furr "Individual Differences in Student Learning Aid Usage" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 528 - 531

variability of percent correct on first try). Then, a process analysis examines how higher and lower performing students navigate and solve a problem, and when in the process they open hints and request solutions. The process analysis was carried out using the R package bupaR¹¹. Using this package, raw data were transformed into event log data, and then mapped to show average transition frequencies and transition times between actions, as a student went about solving a problem. Analyses were carried out at the problem level.

3. CLUSTERING STUDENTS

Students were clustered into two pre-defined groups, using the following metrics that were expected to be related to performance:

- performance: percent correct on first try
- persistence: percent correct on main parts of the problem
- conscientiousness: percent main parts of the problem submitted
- consistency: inverse variability (1 - scaled variability) of percent correct on first try. To correct for variability decreasing as the number of attempted problems increases (students' total number of attempts varied), consistency was calculated from a random sample of 12 attempted problems. The inverse was taken so that the direction of this metric would be higher for better performing students, matching the other metrics.

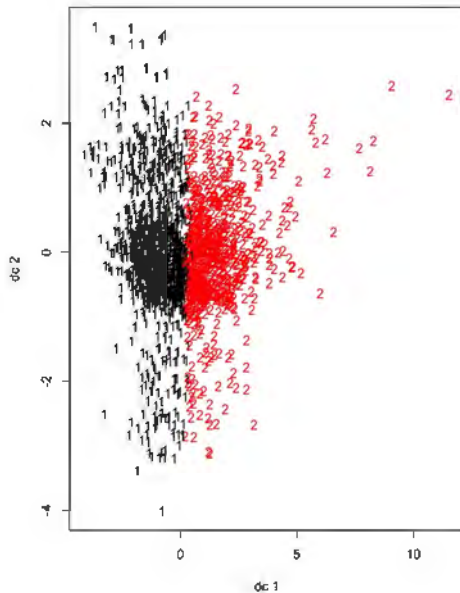


Figure 1. Plot of groups resulting from k-means clustering

Prior to performing k-means clustering, all variables were scaled to have a mean of 0 and a standard deviation of 1. The plot of the groups resulting from k-means clustering is shown in Figure 1, and the group means are shown in Table 1.

Table 1: Group means for higher and lower performers

	Higher performers	Lower performers
performance	0.573	-0.921
persistence	0.560	-0.901
conscientiousness	0.122	-0.196
consistency	0.304	-0.488

4. PROCESS ANALYSIS

Process maps (Figures 2-5) are shown for higher and lower performers, and examining a) relative frequency of actions and b) time between actions at different stages in solving a problem. Relative frequencies of action were calculated by counting the frequency of transitioning from Action A to another action divided by the total number of transitions from action A. Thus, the sum of relative transitional frequencies from one action to all others that followed should be 1. Because problems in the Mastering learning platform are largely designed to be easy (formative assessment), we focused only on student process for more difficult problems. Specifically, using the problem difficulty metric provided by the Mastering learning platform, we excluded problems that fell below the median problem difficulty.

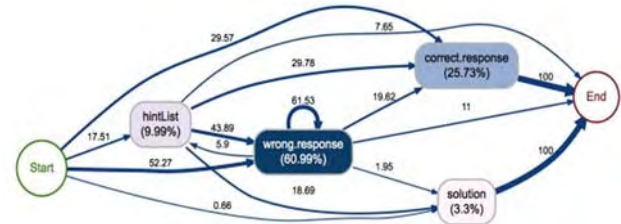


Figure 2. Process map for lower performers

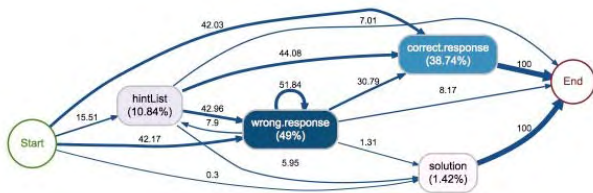


Figure 3. Process map for higher performers



Figure 4. Time between actions for lower performers

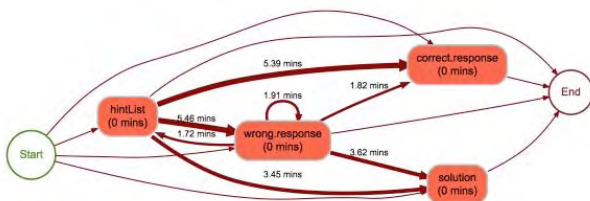


Figure 5. Time between actions for higher performers

Examining first the relative frequency of actions at different stages of solving a problem (Figures 2-3), it appears that lower performers request hints before attempting to answer a problem slightly more often than higher performers (17.51 vs. 15.51). However, higher

performers are *more* likely to request a hint following a wrong response (7.9 vs. 5.9), whereas lower performers are more likely to request a solution following a wrong response (1.95 vs. 1.31) or following a hint (18.69 vs. 5.95). Further, higher performers on average take more time before requesting a solution following a wrong response (3.62 vs. 2.49 minutes) or a hint request (3.45 vs. 2.06 minutes), suggesting they are spending more time thinking over the problem before they request the learning aids. These patterns suggest persistent and conscientious behavior by the higher performers; namely, the higher performers overall seem to use learning aids to solve a problem to completion, rather than simply to get the answer and move on to the next problem.

5. CONCLUSIONS

An exploratory process analysis of learning aid usage revealed different patterns of learning aid use by higher and lower performers. While students were clustered based on metrics related to their responses to problems and not based on any measures of learning aid use, the process analysis found patterns of learning aid use that are suggestive of persistence and conscientiousness in higher performing students. These patterns of use may also lead to better learning, as they suggest deeper processing and better memory encoding, though it may be that higher performers are more highly motivated, which causes both a different pattern of learning aid use as well as better performance. Further work examining the timing of learning aids in terms of before vs. after submitting an incorrect response may disentangle these possibilities, as well as provide evidence-based guidance for when to make hints and solutions available to students in an online learning platform.

6. ACKNOWLEDGMENTS

Our thanks to colleagues at Pearson for their feedback (no pun intended), especially Jinnie Choi, Derek Fay, Anne Pier Salverda, and Jocelyn Sy.

7. REFERENCES

- [1] Siegler, R. S. 1988. Strategy choice procedures and the development of multiplication skill. *Journal of Experimental Psychology: General*, 117, 258-275. DOI=<https://doi.org/10.1037/0096-3445>
- [2] Purpura, J. E. 1998. Investigating the effects of strategy use and second language test performance with high- and low-ability test takers: A structural equation modeling approach. *Language Testing*, 15(3), 333-379. DOI=<https://doi.org/10.1177/026553229801500303>
- [3] Gobbo, C., & Chi, M. 1986. How knowledge is structured and used by expert and novice children. *Child Development*, 1, 221-237. DOI=[https://doi.org/10.1016/S0885-2014\(86\)80002-8](https://doi.org/10.1016/S0885-2014(86)80002-8)
- [4] Bjork, R. A. 1994. Memory and metacognition considerations in the training of human beings. In J. Metcalfe and A. Shimamura (Eds.), *Metacognition: Knowing about knowing* (pp. 185-205). Cambridge, MA: MIT Press.
- [5] Shute, V.J. 2008. Focus on Formative Feedback. *Review of Educational Research*, 78, 153-189. DOI=<https://doi.org/10.3102/0034654307313795>.

- [6] Mathan, S.A. and Koedinger, K.R. 2002. An empirical assessment of comprehension fostering features in an intelligent tutoring system. In S.A. Cerri, G Gouarderes, and F. Paraguacu (Eds.), *Intelligent tutoring systems, 6th international conference ITS 2002: Vol 2363. Lecture notes in computer science* (pp. 330-343), New York: Springer-Verlag. DOI=https://doi.org/10.1007/3-540-47987-2_37
- [7] Mullet, H.G., Butler, A.C., Verdin, B., von Borries, R., and Marsh, E.J. 2014. Delaying feedback promotes transfer of knowledge despite student preferences to receive feedback immediately. *Journal of Applied Research in Memory and Cognition*, 3(3), 22-229.
DOI=<https://doi.org/10.1016/j.jarmac.2014.05.001>
- [8] Kulhavy, R.W. and Anderson, R.C. 1972. Delay-retention effect with multiple-choice tests. *Journal of Educational Psychology*, 63(5), 505-512.
DOI=<http://dx.doi.org/10.1037/h0033243>
- [9] Anderson, D.I., Magill, R.A., and Sekiya, H. 2001. Motor learning as a function of KR schedule and characteristics of task-intrinsic feedback. *Journal of Motor Behavior*, 33(1), 59-67. DOI=<http://doi.org/10.1080/00222890109601903>
- [10] Corbett, A.T. and Anderson, J.R. 2001. Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement, and attitudes. In *Proceedings of ACM CHI 2001 conference on human factors in computing systems* (pp. 245-252). New York: ACM Press.
DOI=<http://doi.org/10.1145/365024.365111>
- [11] Janssenswillen, Gert. 2019. bupaR: Business Process Analysis in R. R package version 0.4.2

N-gram Graphs for Topic Extraction in Educational Forums

Glenn M. Davis*
Stanford University
450 Serra Mall
Stanford, CA
gmdavis@stanford.edu

Cindy Wang*
Stanford University
450 Serra Mall
Stanford, CA
cindyw@cs.stanford.edu

Christina Yuan*
Stanford University
450 Serra Mall
Stanford, CA
cjyuan@stanford.edu

ABSTRACT

Online educational discussion forums allow learners to expand their understanding of key concepts through interacting with peers. However, instructor interaction with forums does not scale well with large and/or busy forums. We introduce graph-based methods for centrality and clustering using post text and user ID in online educational discussion forums. We use the centrality methods HITS and PageRank to identify important and central topics on the forums, and we use the clustering methods Clauset-Newman-Moore (CNM) and spectral clustering to group posts by topic. We demonstrate that these methods can be used at scale to identify important and central topics of discussion that can be brought to the instructor's attention.

Keywords

graph-based methods, natural language processing, online discussion forums

1. INTRODUCTION

Online discussion forums are useful tools for supplementing both online and in-person learning because they provide opportunities to ask questions to instructors remotely and to discuss class topics with peers. However, discussion forum management can be arduous and time-consuming. Post topics and categories typically must be assigned manually by participants or moderators, topic search is often limited to string matching, and meta-scale metrics on forums and communities are not readily available. Thus, despite the scalability of delivering instruction through online courses such as MOOCs (massive open online courses), monitoring and using discussion forums effectively does not scale; rather, instructors and course staff must manually keep track of the forums and attempt to gauge student interest and/or difficulty with course topics.

The structure of these forums, which contain various con-

*All authors contributed equally.

Glenn Davis, Cindy Wang and Christina Yuan "N-gram Graphs for Topic Extraction in Educational Forums" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 532 - 535

nected entities such as questions, answers, users, and topics, relates naturally to graph representations. This paper focuses on using graph constructions of online discussion forums to develop methods for answering two important research questions: 1) What are the most central topics of discussion within the forum? 2) How can we assign categories/topics to individual discussions and posts?

To answer these questions, we introduce a new method for creating *n-gram-based graphs* that contain nodes representing n-gram tokens extracted from posts, connected to nodes representing users and the posts themselves. This graph construction allows us to model the relationship between the contents of each post and the greater overall environment of the discussion forum, including related posts and users. We then use centrality methods to find the most important topics being discussed in the forum and use clustering methods to group communities of posts that discuss similar content.

2. RELATED WORK

[1] used network measures from discussion forums as features for automatically assigning forum participation credit. [5] applied social network analysis to discussion forums from two MOOCs to analyze whether centrality metrics are associated with course performance. These approaches give good baseline methods for graph extraction from educational discussion forum data. However, the network measures that they extracted were fairly limited and focused on participant centrality. We build on their work to explore extraction of more complex relationships and different entities as nodes.

[8] generalized methods from [4] and presented a robust comparative evaluation of different edge weight schemes and centrality measures, as applied to word sense disambiguation. They developed an unsupervised algorithm that constructs a graph given a sequence of words and possible labels (word senses) for each word, where the vertices are labels and the edge weights are dependency scores between word senses. Once the graph is constructed, scores are assigned to vertices using graph-based centrality measures to determine the most likely set of labels for the sequence. We use the idea that textual unit similarity can be used as edges in a graphical representation of a body of text to motivate our construction of n-gram graphs.

3. DATASET

We use the publicly available StackExchange dataset¹. This dataset includes all user-contributed content from over 150 StackExchange discussion forums, with detailed information about user interactions. We use the Posts table for our analysis and extract the columns Body (the body text of a post), Id (the ID number of a post), and OwnerUserId (the ID number of the user who made the post).

We focus on two StackExchange subdomains, Academia and Statistics (referred to from here as Stats). Academia provides a moderately-sized dataset for local CPU computation, with 81,906 posts from 18,640 users. The Stats subdomain provides a more focused and pedagogical approach for our problems, as it is larger and more heterogeneous in both user expertise and topic distribution. However, as the Stats subdomain was too large to compute locally, we extracted the most recent 19,725 posts included in the dataset, which spans from 2017-12-07 to 2018-05-05 and includes 9,094 users.

4. GRAPH CONSTRUCTION

We create two different n-gram-based graph constructions to model our two research questions. In order to find the most central topics of discussion, we model discussion forum data using an *N-gram Graph*, where relationships between n-gram nodes are defined by which users use these n-grams in posts. Then, to assign categories and topics to individual discussions and posts, we create a *Post Graph*, where relationships between post nodes are defined by containing similar n-grams in the text body. The specifics of the graphs are defined below.

4.1 Preprocessing

Since both of our graphs are n-gram-based, we first extracted the most important n-grams, which we call “top terms”, for each post in our dataset. To generate these top terms, we used *Tf-idf* (term frequency-inverse document frequency) weighting over all the n-grams in each post body. This weighting assigns higher importance to the terms in each post in relation with the frequency of the term in the post and the scarcity of the term in other posts. We represent the main topics of discussion for each post by the five terms with the highest *Tf-idf* scores. For both the N-gram Graph and the Post Graph, we create a node for each n-gram that appears in these five top terms for at least one post in the dataset.

For the n-gram graphs created from the Academia subdomain, we represent each post by the five most important unigrams, which extracts terms such as “publish”, “mentor”, and “student”. However, we found that unigrams were not sufficient to capture important topics of discussion in the Stats subdomain, as many technical terms are more than one word in length. Thus, for the Stats subdomain, we instead computed the top terms for each post using the *Tf-idf* scores over both unigrams and bigrams, allowing us to extract top terms such as “bonferroni correction”, “mean”, and “probability measures”.

¹<https://archive.org/details/stackexchange>

4.2 N-gram Graph

To create the N-gram Graph used to model the most important topics addressed in the forums, we first modeled the StackExchange data as a bipartite graph. We created n-gram nodes as described above and user-id nodes for each unique author. We drew an edge between a user-id node and an n-gram node if the user had at least one post where that n-gram appeared as one of the five “top terms”. This bipartite graph, which we call the *User ↔ N-gram Bipartite Graph*, captures the relationship between users and the contents of their posts.

Using the User ↔ N-gram Bipartite Graph, we then folded the graph to create an N-gram Graph that contained only n-gram nodes. In this folded graph, we drew an edge between n-gram nodes if they shared an edge to the same user node in the User ↔ N-gram Bipartite Graph. This yielded a text unit graph similar to those constructed in [4] and [8], except we use network interactions as edges instead of similarity scores. In the N-gram Graph, n-gram nodes that appear in posts by multiple users will have an edge between them. Thus, n-gram nodes corresponding to terms that are discussed by many different users will have a high degree. We then apply centrality methods to identify important and central topics (n-grams) in our network. We apply this procedure to create both an Academia N-gram Graph and a Stats N-gram Graph.

4.3 Post Graph

To create the Post Graph used to determine similarity between posts based on topic, we again modeled the StackExchange data as a bipartite graph. We created n-gram nodes as well as post-id nodes for each unique post. For each post-id node, we drew an edge to the n-gram nodes representing each of its five “top terms”. We call this graph the *Post ↔ N-gram Bipartite Graph*.

Using the Post ↔ N-gram Bipartite Graph, we folded the graph to create a Post Graph that contained only post-id nodes. In this folded graph, we drew an edge between post-id nodes if they shared an edge to at least one n-gram node. We then apply clustering methods to group the post-id nodes into communities defined by their main topics of discussion. We apply this procedure to create both an Academia Post Graph and a Stats Post Graph.

4.4 Graph overview

Table 1 contains node and edge statistics for our graphs. We can see that the Academia graphs are much denser than the Stats graphs both before and after folding.

5. EMPIRICAL FINDINGS

5.1 Centrality: Identifying Top Topics

5.1.1 PageRank

PageRank [2] is an algorithm used to rank nodes in a graph by importance. It treats edges as votes and considers each node to be more important if it has many neighbors. Furthermore, it captures the idea that a “vote” from an important node is worth more, and each edge’s vote is proportional to the importance of the source of its page. We use PageRank on our N-gram Graph to find the most central n-gram nodes.

	Academia				Stats			
Graph	Nodes	Edges	Nodes with degree 0	Nodes with degree > 10	Nodes	Edges	Nodes with degree 0	Nodes with degree > 10
User ↔ N-gram Bipartite Graph	62,765	347,084	0	9,066	66,086	95,267	0	2,308
Post ↔ N-gram Bipartite Graph	126,031	409,530	0	5,776	76,717	98,625	0	1,038
N-gram Graph	44,125	46,984,357	0	38,783	56,992	3,674,828	0	32,718
Post Graph	81,906	55,029,806	61	81,500	19,725	567,729	2,586	14,632

Table 1: Summary statistics for bipartite and folded graphs.

5.1.2 Hubs and Authorities

Hubs and Authorities [6] is an algorithm used to estimate the value of each node’s links to other pages and the value of its own content. These are respectively calculated for each node as its *hub* and *authority* scores. Hub and authority scores are defined via mutual recursion: the algorithm iteratively updates each node’s *hub score* to be equal to the sum of the *authority scores* of each node to which it points, and its *authority score* to be equal to the sum of the *hub scores* of each node from which it is pointed. We use Hubs and Authorities on our User ↔ N-gram Bipartite Graph, with the observation that the n-gram and user-id nodes are analogous to hub and authority pages on the web. That is, we can approximate the value of a user-id node in this graph via its links to frequent and important topics of discussion, and we can approximate the value of an n-gram node as its importance in the forum.

5.1.3 Results

We applied the SNAP² Python implementations of PageRank and Hubs and Authorities on our graphs, in order to rank the centrality of all n-gram nodes. We then identified the highest ranked (i.e., most central) nodes as the most important topics of discussion in each forum. Table 2 shows the top ten nodes identified by PageRank (run on the N-gram Graph) and hub score (run on the User ↔ N-gram Bipartite Graph). Our results show that both centrality measures identify reasonable important topics at face value.

Academia		Stats	
PageRank	Hub score	PageRank	Hub score
paper	user75368	distribution	user8013
student	user53	test	user173082
phd	paper	model	distribution
research	review	matrix	model
journal	student	time	test
review	author	correlation	time
professor	journal	variance	probability
work	supervisor	probability	sample
author	professor	sample	correlation
letter	research	series	variance

Table 2: Top ten topics by PageRank and hub score.

Furthermore, our results validate our formulation of n-grams and users as hubs and authorities of the StackExchange network. The nodes with the highest authority scores are all user-id nodes, and the nodes with top hub scores are all n-gram nodes, with the exception of the four superusers shown

²<https://snap.stanford.edu/>

in Table 2. We manually validated that these identified superusers are the users with the top user-generated reputation score over the time periods observed, suggesting that these users are indeed seen by the community as valuable contributors.

5.2 Clustering: Grouping Posts By Topic

Modularity measures how well a given partitioning of nodes captures separate communities, as compared to a graph with the same number of edges and nodes with random connections. Modularity score Q for an unweighted graph G can be calculated by the expression

$$Q(G, S) = \frac{1}{2m} \sum_{s \in S} \sum_{i \in s} \sum_{j \in s} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

where m is the number of edges in G , $s \in S$ are groups in the partitioning S , i and j are nodes, k_i and k_j are the degree of nodes i and j , A_{ij} indicates whether i and j are connected.

5.2.1 Clauset-Newman-Moore (CNM)

The *Clauset-Newman-Moore* (CNM) algorithm [3] finds communities by greedily optimizing for modularity. Starting with a partitioning of each individual node into its own community, the algorithm repeatedly joins together the two communities that would cause the greatest increase in the modularity score Q , until $n - 1$ joins have been conducted and all nodes belong to a single community. At this point, the algorithm returns the configuration (with a number of communities between 1 and $n - 1$) that produces the highest modularity score.

5.2.2 Spectral Clustering

The *Normalized Cut Algorithm* [7] is a spectral clustering method for partitioning nodes into communities based on the eigenvalues of the symmetric normalized Laplacian. The Normalized Cut Algorithm finds a partition S of the nodes of the graph that gives the smallest normalized cut value: $NCUTS = \frac{cut(S)}{vol(S)} + \frac{cut(\bar{S})}{vol(\bar{S})}$.

To partition the graph into $k > 2$ clusters, we use the *Simultaneous K-Way Cut with Multiple Eigenvectors* modification to the Normalized Cut Algorithm from [7]. To find the optimal number of communities k , we did a search of values of k from 10 to 1000 and computed the modularity score of the communities found for each k value. We selected the k value that gave us the highest modularity score.

CNM		Spectral clustering	
Community (top terms)	Percentage of total nodes	Community (top terms)	Percentage of total nodes
series times group	27.09%	feature dataset value	42.30%
test training validation	23.88%	density normal parameter	5.26%
distribution sample probability	20.43%	training validation data	1.53%
matrix model variables	11.68%	network cost event	1.25%
size power uncertainty	0.81%	series time time series	1.13%
learning rate minutes	0.64%	sample population mean	1.06%
outliers bias percentile	0.49%	distribution normal	1.04%
		distribution probability	

Table 3: Largest communities generated by CNM and spectral clustering

5.2.3 Results

We used the CNM algorithm and the K-Way Cut Normalized Cut Spectral Clustering algorithm on the Stats Post Graph to cluster posts into communities based on similar topics of discussion. CNM generates 66 communities with a modularity of 0.442, and spectral clustering generates 150 communities with a modularity of 0.550. Notably, both algorithms generate community partitions with a modularity score above 0.3, indicating that significant community structure can be detected in our graph [3]. As compared with CNM, spectral clustering produced an optimal partitioning with a larger number of communities and higher modularity score, indicating more consistent partitioning.

Table 3 shows the percentage of nodes in the largest communities found by each clustering algorithm. From this we can see CNM finds four large communities, each with between 10-30% of total nodes, and all other communities contain <1% of the nodes. Spectral clustering finds only one large community that contains 42.30% of the nodes, and more medium sized communities with around 1% of the nodes.

We represent each community found in the Stats subdomain by the three most frequent top terms for posts in the community. Table 3 shows the largest communities generated by both clustering algorithms, and the representative terms for each community. We can see that the most frequent terms that appear in each community are mostly related to one another.

We find that spectral clustering creates a larger number of communities (150 vs. 66), a higher modularity score (0.550 vs. 0.442), and more cohesive communities of smaller size. Thus, we conclude that spectral clustering is more effective at extracting clusters of posts about similar important topics for an educational discussion forum.

6. CONCLUSIONS

We demonstrate that graph methods for computing centrality and clustering can be used to extract important topics and users from an online education-focused discussion forum. Applying our methods across narrow time slices of the discussion forum could allow for instructors to identify topics that require immediate attention in real time, while applying them to a past (archived) offering of the course would highlight important topics for syllabus revision purposes.

Furthermore, identifying clusters of posts about similar topics allows for automatic link generation between posts within

these clusters, and automatic propagation of instructor interventions to all interested parties. Search functionality for both instructors and learners could also be improved through these clusters.

Finally, though n-gram centrality was most salient for the domains we studied, we could similarly compute graph centrality to other entities, such as users and posts. Such methods can be applied to discussion forums that lack a built-in reputation or point system (in StackExchange, user “reputation” and post “votes”), and instructors could use this information to identify potential teaching assistants or moderators, or to allocate course participation grades [1].

7. REFERENCES

- [1] A. Bihani and A. Paepcke. Quantyler: Apportioning credit for student forum participation. In K. E. Boyer and M. Yudelson, editors, *EDM 2018: Proceedings of the 11th international conference on educational data mining*, pages 106–115, Buffalo, NY, 2018.
- [2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.
- [3] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70(6), 2004.
- [4] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004.
- [5] S. Jiang, S. M. Fitzhugh, and M. Warschauer. Social positioning and performance in MOOCs. In *CEUR Workshop Proceedings*, volume 1183, pages 55–58, 2014.
- [6] J. M. Kleinberg. Hubs, authorities, and communities. *ACM computing surveys (CSUR)*, 31(4es):5, 1999.
- [7] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.
- [8] R. Sinha and R. Mihalcea. Unsupervised graph-based word sense disambiguation using measures of word semantic similarity. In *ICSC 2007. International Conference on Semantic Computing*, pages 363–369. IEEE, 2007.

A Data-Driven Approach for Automated Assessment of Scientific Explanations in Science Inquiry

Rachel Dickler
Rutgers University
10 Seminary Place
New Brunswick, NJ 08901
1 (848) 932 - 7496

rachel.dickler@gse.rutgers.edu

Haiying Li
ACT
500 Act Drive
Iowa City, IA
1 (319) 337 - 1000

haiyinglit@gmail.com

Janice Gobert
Rutgers University
10 Seminary Place
New Brunswick, NJ 08901
1 (848) 932 - 0867

janice.gobert@gse.rutgers.edu

ABSTRACT

The Next Generation Science Standards (NGSS [14]) emphasize inquiry practices that students should master including constructing explanations. Most automated methods for scoring scientific explanations, however, do not detect the specific components involved in constructing explanations in the claim, evidence, and reasoning (CER) format, which is commonly used in science classes (cf. [12]). In this study, we expanded and implemented a data-driven, regular expression-based method to automatically score students' written CER explanations across the domains of Physical Science, Earth Science, and Life Science in the intelligent tutoring system, Inq-ITS. We then investigated the generalizability of our method using a new set of testing data. Results of comparisons between human scores and automated scores indicated high performance for the method when applied to the testing data. Analyses showed that the automated scoring for one domain (earth science) had lower performance relative to other domains. Overall, the fine-grained, data-driven, regular expression-based method yielded high accuracy and more reliable scores relative to human scores. Implications for scaffolding CER writing are discussed.

Keywords

automated assessment, regular expressions, science

1. INTRODUCTION

One of the central practices of scientists is constructing explanations of scientific phenomenon. As a result, documents such as the Next Generation Science Standards (NGSS [14]) emphasize that students should master this practice.. Students, however, face many challenges when constructing written explanations (cf., [12]). It is necessary to support students on these difficulties with written explanations in real-time, which is when students benefit most from help [6]. Unfortunately, human scoring of written explanations can be time consuming and subject to rater bias and fatigue [1]. Automated scoring is a solution to these challenges by allowing for real-time evaluation of students' competencies on constructing explanations [18]. However, most automated methods [2, 7, 10, 11, 13, 17] do not detect the specific components involved in CER writing.

Rachel Dickler, Haiying Li and Janice Gobert "A Data-Driven Approach for Automated Assessment of Scientific Explanations in Science Inquiry" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 536 - 539

1.1 Automated Assessment of Scientific Explanations

Automated techniques for scientific explanations vary in terms of the types of algorithms that are used to score students' writing as well as in the particular aspects of students' explanations that are scored. Methods used to automatically score students' explanations include machine learning techniques [13, 10] and natural language processing (NLP) techniques [2, 7, 8, 9, 10, 11, 17]. These methods have been used to score explanations based on the quality of the scientific content of the explanations [2, 7, 10, 11, 13, 17] and the argumentative components within the explanations [8, 9]. In the present study, we focus on methods that applied the NLP method of Regular Expressions (RegEx; [16]) to score study writing.

RegEX [16] involves the application of pattern matching to identify the quality of writing and has been frequently applied to score students' open responses. Researchers at the Educational Testing Service have applied Regular Expressions to score students' scientific writing for the presence of key concepts within conversations [19]. Li et al. [8, 9] applied RegEX in order to assess students' CER explanations in the intelligent tutoring system, Inq-ITS [4, 5]. Specifically, Li et al. [8, 9] developed fine-grained rubrics that captured various sub-components of student CER statements constructed to explain phenomena in the domain of physical science. The RegEX when combined with a linear regression model had high agreement with human raters for each CER sub-component (kappas > .80).

Overall, it is the use of regular expression with if-then algorithms that has shown the best performance in terms of the automated scoring of sub-components of explanations in the claim, evidence, and reasoning format as implemented within the Inq-ITS system [8, 9]. Studies on this method in Inq-ITS to date, however, have been conducted only within the domain of physical science. Researchers have yet to examine whether this automated scoring method would generalize to other domains.

In the present study, we expand the Inq-ITS automated scoring method (i.e., RegEX and if-then algorithms) for CER explanations from the domain of physical science to the domains of life and Earth science.

2. METHOD

2.1 Participants and Sampling Procedure

2,064 sets of claim, evidence, and reasoning (CER) statements (6,195 total statements) were randomly extracted from the intelligent tutoring system, Inq-ITS [4, 5]. 1047 sets of CER responses were constructed in the domain of physical science

(i.e. the Inq-ITS Density Virtual Lab), 481 sets of CER responses were constructed in the domain of life science (i.e. the Inq-ITS Genetics Virtual Lab), and 536 sets of CER responses were constructed in the domain of Earth science (i.e. the Inq-ITS Lunar Phases Virtual Lab). These CER explanations were written by students in grades 6-8 in middle schools in Oregon, Massachusetts, New Jersey, and New Hampshire. Students completed Inq-ITS labs during their regularly scheduled science class periods (see Table 2 for the number of responses for claim, evidence, and reasoning within each driving question in both the training and testing data sets).

2.2 Materials

Inq-ITS [4, 5] is an inquiry intelligent tutoring system for middle school science in the domains of physical, life, and Earth science. Inq-ITS has virtual labs (i.e., microworlds; [3]) in which students carry out inquiry investigations. Each virtual lab has 3-4 different driving questions over the course of four inquiry stages (asking questions/hypothesizing, collecting data, analyzing and interpreting data, and explaining findings in the CER format). All student actions are automatically stored within log files and actions in the first three stages are scored using educational data mining and knowledge engineering techniques (see Sao Pedro et al. [15] for details). Automated scoring of students' CER explanations in the final stage of the lab is the focus of the present study. We randomly selected one virtual lab topic from each science domain (i.e., physical, life, earth) for the present study: Density (Physical Science), Lunar Phases (Earth Science), and Genetics (Life Science).

The Density virtual labs include three driving question activities: (1) how the shape of the container affects the density of the liquid, (2) how the size of the container affects the density of the liquid, and (3) how the type of the liquid affects the density of the liquid. The Lunar Phases virtual labs also included three driving questions: (1) how the percent of the visible Moon illuminated changes, (2) how the duration of lunar orbit changes, and (3) how the percent of the Moon facing the Sun changes. The Genetics virtual labs included three driving question activities: (1) the Mother's F alleles impact the chance of producing the offspring with red fur, (2) the Mother's L alleles impact the chance of producing the offspring with short fur, and (3) the Mother's H alleles impact the chance of producing the offspring with horns. The present study focused on the automated scoring of students' written claim, evidence, and reasoning responses collected within these virtual lab activities (for 9 driving questions in total).

2.3 Rubrics

The rubrics used by human raters were based on a modified version of McNeill et al.'s [12] rubrics (see Table 1 [8, 9]). There were three primary components of the rubric: claim, evidence, and reasoning. The claim component consisted of four sub-components: independent variable (IV), IV relationship (IVR), dependent variable (DV), and DV relationship (DVR). The evidence component consisted of three sub-components: sufficient evidence, appropriateness of data for the independent variable, and appropriateness of data for the dependent variable. The reasoning component included five sub-components: theory or scientific principle, connection between data for the IV, DV, and DVR and the claim, and explaining how the data support (or refute) the claim. This rubric aims to provide more accurate, fine-grained data on how students develop explanations within

the CER framework [8, 9]. Table 1 displays each subcomponent of CER for the Shape-Density lab.

Table 1. Modified rubric from Li et al. [8]

Component	Sub-component		Points
C	IV (Shape)		0-1
	IVR (Conditions)		0-1
	DV (Density)		0-1
	DVR (Same)		0-1
E	Sufficient		0-2
	Appropriate	IV (Mass + Volume)	0-1
		DV (Density)	0-1
R	Theory		0-2
	Connection		0-1
	Data	IV/IVR	0-1
		DV	0-1
		DVR	0-1

Note. IV = independent variable. IVR = how IV was changed. DV = dependent variable. DVR = how DV was changed.

The rubrics for each activity were modified based on the specific content of each activity. Therefore, the structure of each rubric was the same in terms of the sub-components that were scored, but the specific expressions that were captured for each activity (i.e., IV = shape for the Shape-Density activity) differed based on the content of the virtual lab activity. The generation of specific rubrics for the three driving questions within each microworld took about 10-30 minutes [9].

2.4 Human Scoring

One senior researcher and three research assistants graded students' CER statements. All raters received extensive training on how to use the rubrics with practice responses (not included in the training or testing set). Raters continued to receive training until inter-rater reliabilities on practice responses were above kappa of .80. Prior to scoring, the data was randomly split into two sets: training data and testing data.

For the training data, the first activity from each topic was scored by two randomly assigned human raters independently. Rubrics were reviewed with raters prior to scoring. The inter-rater reliability was above Pearson correlations of .80 for each CER sub-component for each of the first activities for each topic, so disagreements were discussed and resolved, and agreed upon score were used as the "human scores" for the generation of automated algorithms and reliability analyses. The training data for the remaining two activities within each topic were then each scored by one rater independently. For the testing data (i.e., 60 sets of CER responses for each activity), two raters were randomly assigned to independently score responses for each activity. After scoring of the testing data was completed, human-human interrater-reliabilities were computed for each activity (see Table 2), disagreements between human raters were identified and resolved through discussion, and agreed upon scores were used for analyses (i.e., human scores).

2.5 Automated Approach

In prior studies [8, 9] we manually developed regular expressions to detect patterns for each subcomponent of students' claim, evidence, and reasoning (CER) statements within the Shape-Density activity based on a corresponding rubric used by human raters. In the present study, we kept the basic RegEX patterns for each subcomponent for CER, but changed the key expressions as concepts changed in different

activities. It took approximately 3 hours total to generate RegEX for each statement type (C, E, or R) for each activity. The structure of Inq-ITS (i.e., enabling students to first construct their claim and evidence within interactive widgets before writing their claim and evidence) affords identification of the types of potential phrases and structures that students may use in their CER explanations.

Phrases that were identified as semantically similar to the components outlined in the human scoring rubrics were used as the basis for pattern matching in the RegEX. We then used a data-drive approach to modify the expressions based on the words or expressions that students used in their responses in the training data set. For example, the IV in the L Alleles activity in Genetics is “L alleles.” When we looked through students’ responses though, we found some alternative expressions, such as “fur length alleles” and “alleles for hair length.” Thus, we modified the original RegEX (i.e., “\b[Ll]b”) accordingly. We then compared the automated scores with the human scores, looked through the responses to identify where there was a disagreement, noted the variations in the expressions, and then modified the RegEX. We repeated this cycle until we achieved high performance, with correlations above .95 or even 1.00.

Implementation of the RegEX in Python allowed for identifying whether particular subcomponents were represented in students’ statements or not, as well as the accuracy of subcomponents included within students’ statements [8, 9]. In particular, the presence of components at each level (0 points = incorrect; 0.5 = partially correct/present; 0.8 = mostly correct/present; 1 = correct/present) is identified and assigned to a corresponding sub-component. We then used the generated RegEX and algorithms to examine the performance of the RegEX and if-then algorithms when applied to a testing data set (i.e., 60 responses from each activity). This study used Pearson correlations - a commonly used metric [10] - to evaluate the accuracy of the automated scoring.

3. RESULTS

We first examined performance of the RegEX at an aggregated level (across all data sets), then by topic, and finally by driving question within topics. Our analyses revealed that there was extremely high agreement for CER between the humans and automated computer scores when examining scores aggregated across all data sets. The correlations in the training data sets were .99 for claims, .98 for evidence, and .97 for reasoning. The correlations in the testing data sets were .90 for claims, .94 for evidence, and .86 for reasoning.

There was also high extremely high agreement when breaking down claim, evidence, and reasoning scoring by topic (i.e., density, lunar phases, and genetics) with all human-computer correlations above 0.95 for the training data and all correlations in the density and genetics labs above 0.90 for the testing data. However, the accuracy of the automated method was relatively low for the reasoning component of lunar phases ($r = 0.73$). It is important to look at each activity within each topic of lunar phases to understand why correlations were lower for reasoning as compared to the other two virtual lab topics.

Finally, we examined the correlations between human and automated computer scores for each activity within each topic. For the training data, the results showed that all correlations were above 0.95 (see Table 2). For the testing data, all correlations were above 0.80 except for the claim for the Lunar

Phases percent of the visible moon illuminated activity ($r = 0.75$) and the reasoning for the Lunar Phases duration of Lunar Orbit ($r = 0.76$). To further understand these discrepancies, we conducted fine-grained analyses using confusion matrices for the claim sub-components for the percent of the visible moon illuminated activity and the reasoning sub-components of the duration of lunar orbit activity.

Table 2. Correlations by virtual lab activity

Activity	CER	Training		Testing		
		H-C	N	H-H	H-C	N
Density						
Shape - Density	C	0.99	293	0.97	0.97	60
	E	0.98	293	0.95	0.97	60
	R	0.97	293	0.93	0.86	60
Amount -Density	C	0.99	305	0.97	0.95	60
	E	0.95	305	0.94	0.86	60
	R	0.97	305	0.84	0.96	60
Type - Density	C	0.99	269	0.97	0.97	60
	E	0.99	269	0.98	0.97	60
	R	0.96	269	0.90	0.96	60
Lunar Phases						
Visible Moon - LP	C	0.95	140	1	0.75	60
	E	0.97	140	0.97	0.95	60
	R	0.98	140	1	0.81	60
Duration -LP	C	0.99	115	1	0.91	60
	E	0.99	115	1	0.93	60
	R	0.96	115	1	0.76	60
Moon Facing Sun -LP	C	1	101	1	0.95	60
	E	1	101	1	0.89	60
	R	0.98	101	1	0.81	60
Genetics						
F allele-Genetics	C	0.99	100	0.90	0.89	60
	E	0.98	100	0.89	0.93	60
	R	0.98	100	0.83	0.88	60
L allele-Genetics	C	1	100	0.90	0.91	60
	E	0.98	100	0.90	0.95	60
	R	1	100	0.79	0.94	60
H allele-Genetics	C	1	99	0.84	0.93	60
	E	1	99	0.91	0.96	60
	R	1	99	0.87	0.90	60

Note. C = Claim. E= Evidence. R= Reasoning. H-C = Human-Computer. H-H = Human-Human. LP = Lunar Phases.

For the lunar phases visible moon illuminated activity, there were several disagreements between the humans and computer for the claim sub-components of: the change in the independent variable (IVR; $r = 0.35$) and the dependent variable (DV; $r = 0.48$). Further analyses using the confusion matrix for the claim IVR scores assigned by the human and computer revealed that there were 14 cases where the computer gave credit (i.e., 1 point) for the presence of the IVR, but the humans did not).The

confusion matrix for the claim DV scores assigned by the human and computer revealed that there were 10 cases where the humans gave students credit (i.e., 1 point) for the DV, but the computer did not.

For the lunar phases duration of lunar orbit activity, there were several disagreements between the humans and computer for the reasoning sub-components of: theory ($r = 0.29$), explaining data for the DV (Data-DV; $r = 0.66$), and explaining data for the change in the DV (Data-DVR; $r = 0.74$). In terms of the reasoning theory scores assigned by the human and computer, a confusion matrix revealed that there were 24 cases where the computer gave students credit (i.e., 2 points) for theory but the humans did not. The confusion matrix for the reasoning Data-DV subcomponent showed that there were 7 cases where humans gave students credit (i.e., 1 point) and the computer did not give any credit. The confusion matrix for the Data-DVR component showed that the biggest differences in scoring were in 4 cases where the computer gave students' credit (i.e., 1 point) but the humans did not give any credit.

4. DISCUSSION

Overall our findings show that the automated scoring method in Inq-ITS is extremely high performing when applied to new data across different domains (i.e., physical, earth, and life science). In particular, the development of RegEX and if-then algorithms using our iterative method was both efficient and effective in terms of being able to capture student performance on explanations in the CER format at a fine-grained level. Our method therefore extends upon other NLP scoring techniques for scientific explanations (i.e., that captured only the linguistic quality or content of explanations; [2, 7, 10, 11, 13, 17]) because it is able to accurately capture students' competencies on the argumentative components involved in constructing explanations in the CER format. In the future, it would be valuable to explore other data processing methods and their potential for even greater efficiency for scoring written CER.

The findings of this study are promising in terms of being able to accurately assess student performance in real-time in multiple domains at a fine-grained level. Specifically, this automated method can now be implemented within the intelligent tutoring system, Inq-ITS, in order to capture student performance as they are constructing their CER explanations. Additionally, this fine-grained scoring can be used to detect student difficulties with particular components of CER statements and, as a result, allow for real-time scaffolding at a fine-grained level in real time. This automated scoring can also be used to alert teachers at an actionable level using Inq-Blotter, our teacher dashboard.

5. REFERENCES

- [1] Bejar, I. I. 2012. Rater cognition: implications for validity. *Educ. Meas.* 31 (Sept. 2012), 2-9.
- [2] Crossley, S. A., Kyle, K., Davenport, J. L., and McNamara, D. S. 2016. Automatic assessment of constructed response data in a chemistry tutor. In: *EDM* (June. 2016), 336-340.
- [3] Gobert, J. 2015. Microworlds. In: *Encyclopedia of Science Education*, 638-639.
- [4] Gobert, J.D., Sao Pedro, M., Raziuddin, J., Baker, R.S. 2013. From log files to assessment metrics: measuring students' science inquiry skills using educational data mining. *J. Learn. Sci.* 22 (2013), 521-563.
- [5] Gobert, J. D., Baker, R. S., and Sao Pedro, M. A. 2014. Inquiry skills tutoring system. (Jan. 2014). US Patent no. 9,373,082. Issued Jan. 29th, 2014.
- [6] Koedinger, K. R., and Anderson, J. R. 1998. Illustrating principled design: the early evolution of a cognitive tutor for algebra symbolization. *Interactive Learning Environments* 5 (1998), 161-180.
- [7] Li, H., Shubeck, K., and Graesser, A. C. 2016. *Using Technology in Language Assessment*. Bloomsbury Academic, London, UK.
- [8] Li, H., Gobert, J., and Dickler, R. 2017. Automated assessment for scientific explanations in on-line science inquiry. In: *EDM*. Wuhan, China (June. 2017), 214-219.
- [9] Li, H., Gobert, J., and Dickler, R. 2017. Analyzing the sub-skills underlying students' scientific claims, evidence, and reasoning during inquiry. In: *27th Annual Meeting of ST&D*. Philadelphia, PA (July. 2017).
- [10] Liu, O. L., Rios, J. A., Heilman, M., Gerard, L., and Linn, M. C. 2016. Validation of automated scoring of science assessments. *J. Res. Sci. Teach.* 53 (Jan. 2016), 215-233.
- [11] McNamara, D.S., Graesser, A.C., McCarthy, P.M., Cai, Z. 2014. *Automated evaluation of text and discourse with Coh-Metrix*. Cambridge University Press, New York.
- [12] McNeill, K., Lizotte, D.J., Krajcik, J., and Marx, R.W. 2006. Supporting students' construction of scientific explanations by fading scaffolds in instructional materials. *J. Learn. Sci.* 15 (Nov. 2006), 153-191.
- [13] Moharreri, K., Ha, M., and Nehm, R. H. 2014. EvoGrader: an online formative assessment tool for automatically evaluating written evolutionary explanations. *Evol.* 7 (Aug. 2014), 15.
- [14] Next Generation Science Standards (NGSS) Lead States. 2013. *Next Generation Science Standards: For States, by States*. The National Academies Press, Washington, DC.
- [15] Sao Pedro, M. A., Baker, R. S., Gobert, J. D., Montalvo, O., and Nakama, A. 2013. Leveraging machine-learned detectors of systematic inquiry behavior to estimate and predict transfer of inquiry skill. *User Model. User-Adap.* 23 (Mar. 2013), 1-39.
- [16] Thompson, K. 1968. Regular expression search algorithm. *Commun. ACM.* 11 (June. 1968), 419-422.
- [17] Weston, M., Parker, J., and Urban-Lurain, M. 2013. Comparing formative feedback reports: human and automated text analysis of constructed response questions in biology. In: *NARST* (Apr. 2013).
- [18] Williamson, D. M., Xi, X., and Breyer, F. J. 2012. A framework for evaluation and use of automated scoring. *Educ. Meas.* 31 (Mar. 2012), 2-13.
- [19] Zapata-Rivera, D., Lehman, B., Jackson, T., and Liu, L. 2019. Refining conversation-based assessments of science inquiry skills. In: *AERA*. Toronto, Canada (Apr. 2019).

Design and Deployment of a Better University Course Search: Inferring Latent Keywords from Enrollments

Matthew Dong
UC Berkeley
Berkeley, CA, USA
mdong@berkeley.edu

Run Yu
Wuhan University
Wuhan, China
run.yu@whu.edu.cn

Zachary A. Pardos
UC Berkeley
Berkeley, CA, USA
pardos@berkeley.edu

ABSTRACT

Liberal arts universities possess a vast catalog of courses from which students can choose. The common approach to surfacing these courses has been through traditional keyword matching information retrieval. The course catalog description used to match on may, however, be overly brief and omit important topics covered in the course. Furthermore, even if the description is verbose, novice students may use search terms that do not match relevant courses, due to their catalog descriptions being written in the specialized language of a discipline outside of their own. In this work, we design and user test an approach intended to help mitigate these issues by augmenting course catalog descriptions with topic keywords inferred to be relevant to the course by analyzing the information conveyed by student co-enrollment networks. We tune a neural course embedding model based on enrollment sequences, then regress the embedding to a bag-of-words representation of course descriptions. Using this technique, we are able to predict potentially relevant words that are not in a course's description and surface these words through a real-world recommendation platform.

Keywords

Course search, Inferred keywords, Latent topics, Course2vec, Skip-gram, Higher education, Recommender systems

1 Introduction

The course catalog is often the first resource consulted by current and prospective students when wanting to familiarize themselves with and explore the topical offerings of a university. With many universities offering thousands of distinct courses over the span of several years, browsing through the description of each is untenable. Instead, classical information retrieval (i.e., search) using keyword matching is now offered at many, but not all, institutions. A keyword matching approach; however, is only as good as the words the description contains and the users' ability to use query terms that will match. Many course descriptions can be overly brief, omitting topical terms from the descrip-

Matthew Dong, Run Yu and Zachary Pardos "Design and Deployment of a Better University Course Search: Inferring Latent Keywords from Enrollments" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 540 - 543

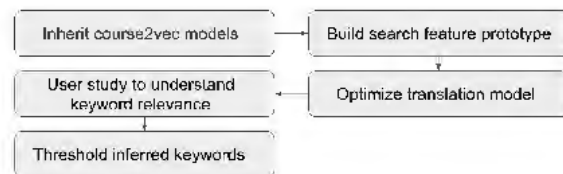


Figure 1: Design process for the enhanced search

tion that are nevertheless contained in the course. Furthermore, for novice students, it can be difficult to gauge the similarity of courses in different departments because of the superficial differences in how different disciplines describe the same material.

In this paper, we seek to mitigate the shortcomings of topic omission and non-standardized keywords across disciplines in catalog descriptions by leveraging the regularizing power of machine learned embeddings. We apply neural embedding models to historic sequences of student course enrollments in order to embed courses into a space regularized by abstract features, or concepts, associated with courses. We then regress from this space to the space of course descriptions in order to add semantics to the course vector space. These semantics become the keywords which can be added to an enhanced university course search. Our approach is closer to the user experience of an information system but using machine learning techniques more commonly seen in collaborative-based models. This adding of keywords to an object could be framed as a form of topic modeling. Motz et al. [3] provide an approach in this vein most relevant to ours, in which they use students' course enrollments as a signature with which to learn themes of studying using Latent Dirichlet Allocation (LDA). We substitute LDA with the more contemporary machine regularization of skip-gram models [2] and take the work further by conducting a user study ($N = 75$) in which students at a university were asked to rate relevancy of keywords to courses they had taken, generated from the embedding model and other baselines (e.g., random within and outside of description words). Measuring the degree to which our model's inferred keywords correlate with student perceptions of relevance, the results suggest a probability threshold above which predicted out of course description keywords can be chosen and be expected to be more relevant to students than the random within description baseline. Furthermore, we close the research loop by integrating this modeling process into a larger design scheme (Figure 1) leading to the deployment of this enhanced course



Figure 2: A prototype of the course search feature before model tuning and user testing

search feature in a live course recommendation system linked to by the campus’ Office of the Registrar website.

2 Models

Our approach to generating inferred course keywords comprises of three fundamental modeling elements: (1) a vector representation of courses learned from enrollment histories (2) a bag-of-words representation of course catalog descriptions (3) a model that translates from the enrollment-based representation to the catalog-based representation. This is essentially a machine translation, not between languages [1], but between a course representation space formed from student enrollment patterns and a semantic space constructed from instructors’ descriptions of the knowledge imparted in each course.

2.1 Course2Vec

The course2vec [4] model involves learning distributed representations of courses from students’ enrollment records throughout semesters by using a notion of an enrollment sequence as a “sentence” and courses within the sequence as “words”, borrowing terminology from the linguistic domain. For each student s , a chronological course enrollment sequence is produced by first sorting by semester then randomly serializing within-semester course order. Then, each course enrollment sequence is trained on like a sentence in a skip-gram model. In language models, two word vectors will be cosine similar if they share similar sentence contexts. Likewise, in the university domain, courses that share similar co-enrollments, and similar previous and next semester enrollments, will likely be close to one another in the vector space. Course2vec learns course representations using a skip-gram model by maximizing the objective function of context prediction over all the students’ course enrollment sequences.

It is important to stress that our method of producing a course vector from enrollments (i.e., course2vec) does **not** use any course description information. It is based only on sequences of course IDs, with no natural language used. The generalizing principal is that patterns of student collective course taking can produce representations of courses containing abstract concepts of relevance to student course search. The trick to exploiting this is to associate these abstract concepts with concrete keywords, accomplished by the translation model, explained in the section after the next.

Table 1: Course Keyword Groups Example	
Course: STAT 135 - Concepts of Statistics	
Course Description: A comprehensive survey course in statistical theory and methodology. Topics include descriptive statistics, maximum likelihood estimation, non-parametric methods, introduction to optimality, goodness-of-fit tests, analysis of variance, bootstrap and computer-intensive methods and least squares estimation. The laboratory includes computer-based data-analytic applications to science and engineering.	
Model Sorted (All): regression, statistics, random, statistical, estimation	
Model Sorted (Description): statistics, statistical, estimation, variance, tests	
Model Sorted (Non-Description): regression, random, real, linear, discrete	
Random (Description): course, engineering, includes, methods, computer-based	
Random (All): diverse collection, topics problems, year credit, planning research, user interfaces	

2.2 Bag-of-Words Representation

We represent course catalog descriptions using the simple but indelible approach of bag-of-words and its variants. To create a course description vector, the length of the number of unique words across all items serves as the dimension of the vector, with a non-zero value if the word in that vocabulary appears in the description. We experiment with the description vector as binary, tf-idf, as well as a custom weighting scheme such as tf-bias that controls the granularity of keywords represented.

2.3 Translation Model

Our premise is that there are useful concepts learned in the embedding of course2vec, but these concepts left in abstract vector form contain no explicit semantics. To associate the patterns learned in course2vec with semantics, we apply a translation from the course2vec vector to its respective natural language course description vector.

We use a multinomial logistic regression to conduct this mapping, where the skip-gram based course vectors are used as input and the corresponding descriptions of every course as bag-of-word encodings are the multi-hot labels being predicted. After this model is trained, the probabilities of each word in the vocabulary belonging to a skip-gram course vector can be computed by consulting the softmax probability distribution over the entire vocabulary. Using this probability distribution, it is now possible to find the high probability words predicted based on course2vec which are NOT in the course description. These words can subsequently serve as inferred keywords in our enhanced course search.

Logistic regression is used to represent translation between languages because the spaces being translated to and from are linear vector spaces (skip-grams have no non-linear activations). However, in case the translation between spaces in the course domain is a non-linear one, we also evaluate a single hidden layer neural network with non-linear activation as a candidate translation model in our optimization experiments.

Step 2: Rate each keyword's relevance to the course content, based on the following scale (this scale is only for your personal reference):

1 Not Very Relevant 2 Not Very Relevant 3 Not Very Relevant 4 Not Very Relevant 5 Very Relevant

Statistics 105: Concepts of Statistics
You took the course during Spring 2019 with:

Keyword	Relevance
11	1 2 3 4 5
12	1 2 3 4 5
13	1 2 3 4 5
14	1 2 3 4 5
15	1 2 3 4 5
16	1 2 3 4 5
17	1 2 3 4 5
18	1 2 3 4 5
19	1 2 3 4 5
20	1 2 3 4 5

Figure 3: Keyword rating form for given course

3 USER STUDY

Through offline model selection based on 144 experiments attempting to optimize heuristics expected to correlate with relevancy ratings, we settled on a regression model and corresponding hyperparameters that maximized our custom model evaluation metric. Following the experiment driven model selection, we follow up with a human judgment evaluation to better gauge how the model results are aligned with students' perception of relevance. A user study was conducted during which students were asked to rate keywords belonging to five different groups:

1. *Model Sorted (All)*: Top five overall keywords as predicted by the model.
2. *Model Sorted (Description)*: Top five words in the description in order of likelihood as predicted by the model.
3. *Model Sorted (Non-Description)*: Top five words not in the description in order of likelihood as predicted by the model.
4. *Random (Description)*: Five random words from within the description.
5. *Random (All)*: Five random words across all collective descriptions.

An example of these keyword groups for a particular course are shown in Table 1. The Random (All) words represent a baseline relevancy score. We expect the description groups to perform much better than this baseline and desire that the model predicted non-description words are also better than randomly selected words. The random (description) group provides the second benchmark to compare our model sorted non-description group to, quantifying how much value our enhanced search proposes to add on top of the catalog description.

3.1 Study Design

Undergraduates were recruited from popular university associated Facebook groups to participate remotely in exchange for a \$10 Amazon gift certificate. Study participants logged into the main *AskOski* recommender site using their university credentials in order to access the survey. Figure 3 shows the rating form for one course with its corresponding unique keywords from each of the five groups randomly shuffled. We intentionally did not show the description and

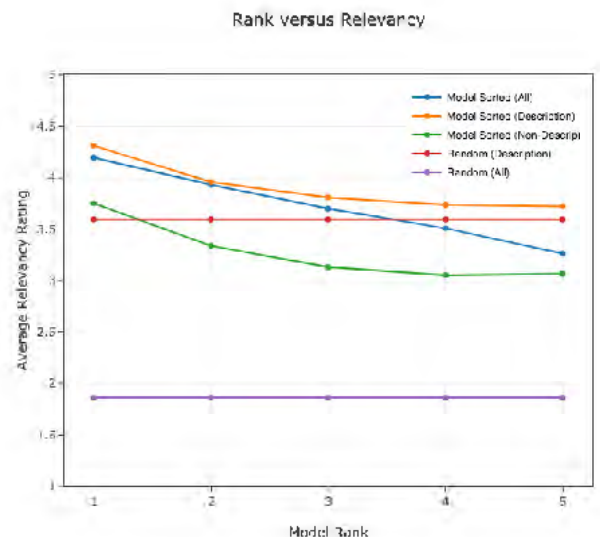


Figure 4: Keyword group rank vs relevancy

requested students to not look them up and rate solely on their experience with the class to prevent bias in keyword ratings whereby a student may be tempted to simply rate a word as relevant only if it appeared in the description.

For every keyword, students were asked for their five point Likert scale agreement with the following statement: *This keyword is relevant to the course*, where a score of 1 corresponded with *Not Relevant At All* and a score of 5 corresponded with *Very Relevant*. A total of 75 students participated in our study, rating a total of 8,355 keywords.

3.2 Results

The average student relevancy ratings of keywords from each of the five groups is shown in Figure 5. All three Model Sorted groups, and the Random (Description) group, scored between a 3 (neutral) and 4 (relevant) in keyword relevance. Selecting keywords at random from the entire vocabulary, Random (All), scored a 1.836 (below "Not Very Relevant"), representing students' lower bound for perception of relevance. All pairwise differences between keyword groups were statistically significantly reliable at $p < 0.005$, after applying a Bonferroni correction for multiple (10) Wilcoxon rank sum tests, except between Model Sorted (All) and Random (Description) groups which was not statistically separable ($p = 0.019$).

The benefit of the model-based approach in terms of improving relevance of chosen keywords can be quantified by the difference in ratings between the random within-description selection group, Random (Description) - 3.612, and the model-based within-description selection group, Model Sorted (Description) - 3.916. A breakdown of the proportion of each rating level by group can be seen in Figure 5. The majority (51%) of Model Sorted (Description) keywords received a 5 rating (Very Relevant), compared to Random (Description), for which 42.1% were Very Relevant. Model Sorted (Non-Description) has a much lower proportion of Very Relevant ratings (31.5%), but still considerably higher than the Random (All) baseline, with 7.3%, and with 62.3% of keywords in its group receiving the lowest relevancy rating as compared with Model Sorted (Description), that received 20.6% Not Relevant ratings.

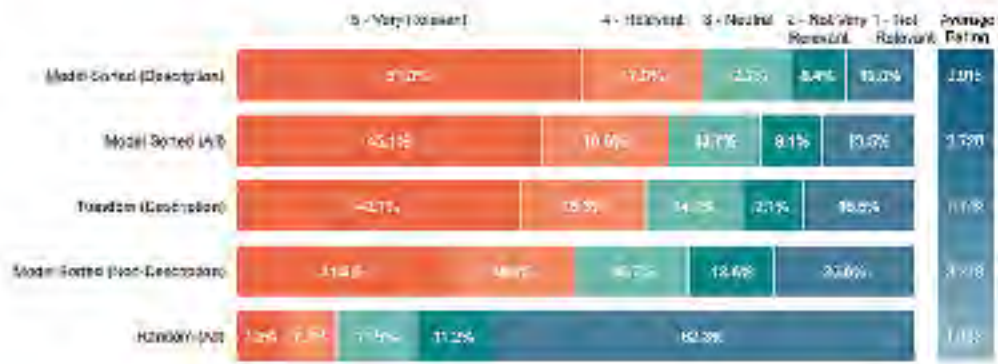


Figure 5: User study relevancy ratings by keyword group

The way in which student relevancy ratings played out with respect to the within-group ranking of the keyword based on model probability is shown in Figure 4. The average relevancy rating (y-axis) by rank (x-axis) is plotted for each of the three model-based approaches. Since the two random models do not involve any model probabilities, they also are not associated with a rank. Therefore, they are represented in the plot as horizontal lines corresponding to their averages. The Model Sorted (All) trend shows the highest average ratings at rank 1, followed by an apparent asymptote down to just above the average random within-description level. Differences in ratings between these two at each rank level are statistically significantly reliable except at ranks 3 and 4. The Model Sorted (Non-Descrip) trend is initially above Random (Description) at rank 1, but then dips down and asymptotes to a Neutral average ratings value of 3.

A premised benefit of the predictive model was to surface relevant keywords that are not in a course’s description (Non-Descrip). If we were to highlight inferred keywords, we would like to show only keywords that are “better” than words chosen randomly from the description, or at least not show words statistically significantly worse. The Model Sorted (All) ratings are statistically reliably higher than Random (Description) at ranks 1 and 2. We use this information to tailor our strategy for when and how many inferred keywords to display in the production version of our enhanced course search feature.

3.3 Selecting keywords to display in search

With an improved understanding of the relevancy of the model’s predicted keywords, we discuss how to leverage this information towards improving the search feature by updating our inferred keyword selection criteria. In the prototype, the criterion was to always display the top 10 model keywords, which did not exclude words in the description. We continue to not restrict the display of keywords from the description, as showing them could serve the added benefit of a topic category source for reference. Thus, we choose Model Sorted (All) as the focus of this analysis.

We leverage the observation that Model Sorted ratings correlate with rank to investigate how well the underlying model probabilities of those words correlate with student relevancy ratings. If there is a correlation, then the probabilities, along with a threshold, could be used to dynamically determine which words should be included as inferred keywords on a per course basis. To conduct an analysis comparing model

probabilities to user ratings, we normalize these two sets of ratings using Z-scores and then average them by Model Sorted rank. We find a substantive correlation between probability and rank and would like to choose a threshold of probability from Model Sorted (All), such that all keywords with that probability or above can generally be expected to produce keywords perceived by students to be more relevant, on average, than a word chosen at random from the description. The analysis in the previous section (Fig 4) found that user relevancy ratings for Model Sorted (All) were significantly higher than Random (Description) at ranks 1 and 2. Therefore, we use the probability at rank 2 as the cut-off. Using this probability cut-off, we find 4.32 total words on average expected to be displayed for each course, with 2.33 within-description words and 2.00 non-description words surfaced on average within these semantics.

4 Conclusion

We explored surfacing novel, searchable semantics of a course using an embedding of courses informed by course selection histories, and supported our methodology through a user study to evaluate the relevancy of these keywords. Our experiment contributes both methodologically to the use of embeddings to surface latent semantics and to the design of data-driven information systems in educational settings. Our process of interface prototyping, followed by offline model optimization, user testing, and incorporation of study findings into the production software system can also serve as a design model and guide for other technologies to tune EDM analyses towards better student experiences.

5 References

- [1] T. Mikolov, Q. V. Le, and I. Sutskever. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168, 2013.
- [2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
- [3] B. Motz, T. Busey, M. Rickert, and D. Landy. Finding topics in enrollment data. In *Proceedings of the 11th International Conference on Educational Data Mining*, 2018.
- [4] Z. A. Pardos and A. J. H. Nam. A map of knowledge. *CoRR*, abs/1811.07974, 2018.

Visualizing Learning Performance Data and Model Predictions as Objects in a 3D Space

Bruno Emond
National Research Council Canada
1200 Montreal road, Ottawa, ON
Canada, K1A 0R6
1 (603) 993-0154
bruno.emond@nrc-cnrc.gc.ca

Julio J. Valdés
National Research Council Canada
1200 Montreal road, Ottawa, ON
Canada, K1A 0R6
1 (603) 993-0257
julio.valdes@nrc-cnrc.gc.ca

ABSTRACT

This poster/demonstration presents our preliminary efforts to explore learning data and model predictions by mapping them into objects in a three-dimensional space. Our work is related to the area of visual analytics, where machine learning or other analytic techniques are combined with interactive data visualization to promote sense making and analytical reasoning. The demonstration consists of interactive VRML models generated from the DataShop Geometry9697 dataset. The initial results indicate that in spite of some limitations the approach allowed to identify from only the observations distribution specific knowledge components that could be targeted for model refinement.

Keywords

Visual analytics; Dimensionality reduction; Closing the loop between research and practice.

1. INTRODUCTION

The visual inspection of learning curves plays an important role in the discovery of anomalies in learning models [1,10]. In principle, learning curves exhibit a smooth decrease of error rates as a function of the increase number of learning opportunities [11]. When anomalies are visually detected (flat curves, curve spikes, slope increases), they are interpreted as hidden non-obvious knowledge components (KC) [2] that need to be made explicit in the model. Model refinements based on either logistic regression models or hidden Markov models, assume that refinements take place by modifying knowledge component assignments to learning items. However, visual inspection has its limitations, and alternative means of exploring alternative model refinements through computations and model fitness assessments is an active area of research [2].

However, given the need to close the loop between research and practice in order to better embed evidence-based education and training, visual analytics might provide an interesting approach to explore. Visual data inspection is a useful operation at the initial stages of data analysis to provide insights about regularities,

oddities, changes, and trends. But as visual analytics human-in-the-loop process models suggest [12,13], visual information is also important at all stages of data mining refinements [15,16].

Our work is related to the area of visual analytics, where machine learning or other analytic techniques are combined with interactive data visualization to promote sense making and analytical reasoning [5]. Our exploration of visual analytics for educational data mining is preliminary and focuses on initial visual representations such as the relationship between: 1) error rates and learning opportunities, 2) observed and predicted error rates for knowledge components, and 3) observed and predicted error rates for knowledge components as a function of learning opportunity phases. The demonstration consists of interactive Virtual Reality Modeling Language (VRML) models generated from the DataShop 'Geometry Area (1996-97)' dataset [7] using unsupervised, and nonlinear methods. VRML is simply used as a format to support data visualisation, other formats are also possible such as X3D, FBX, or OBJ. The file format for encoding 3D models is independent from the mapping method of an original data set of n -dimensions to a three-dimensions space.

Prior to displaying the data for visual inspection using VRML, the method reduces the multidimensional attribute values of the data source to a three-dimension geometric space exposing object proximity, known object links, and object clusters. The method seeks distance/dissimilarity preservation between the original high-dimensional space and a target low-dimensional space, so that any relationship found between these patterns and any other external variable is due to the existence of interdependencies between that variable and the original descriptors [15]. The purpose is to make all distances comparable, regardless of how many attributes are used. For example, objects 1 and 2 may have 5 attributes in common, but 1 and 3 may have 10. Normalizing by the number of common attributes eliminates the differences when comparing distances computed with different numbers of attributes.

As an exploratory data analysis method, mapping a data set into a three-dimensional space has as a primary objective to reveal patterns in the data that could be worth of further investigation. In spite of the fact that visual inspection is not exempt of biases [4], the dimensions reduction method ensures that the loss of information on all objects pairs similarity distances in the mapping from N to 3 dimensions is minimal, preserving the relative proportional distances as much as possible. For the analysis performed in the current paper, the three-dimensional space was computed using the Sammon's nonlinear mapping method [14].

In similarity with t-SNE [9], data points represented in the space have to be interpreted in their spatial relationships with the other

Bruno Emond and Julio J. Valdés "Visualizing Learning Performance Data and Model Predictions as Objects in a 3D Space" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 544 - 547

data points, leaving undetermined the meaning of individual point values in relation to the space axis. One advantage of the mapping to three dimensions in comparison to a mapping to two dimensions, is the reduction of information loss while preserving visual interpretability.

2. ANALYSIS

The next sections explore the application of visual analytics methods to the representation of relation between: 1) error rates and learning opportunities, 2) observed and predicted error rates for knowledge components, and 3) observed and predicted error rates for knowledge components as a function of learning opportunity phases. The DataShop Geometry9697 dataset [7] is used for the analysis, but more specifically the original model which has 15 knowledge components. Other models from the same dataset range in number from 1 to 15 with a median of 12 knowledge components [8].

2.1 Opportunities As Objects

Figure 1 shows the DataShop learning curve for the opportunities averaged over students. One can easily see that the observations in red and the prediction in blue show a reduction of error rates as the number of opportunities increases.

For our first analysis, we were exploring if the 3D modelling method would show a similar trend and structure. The first analysis used an input data matrix containing 74 opportunity objects (37 observations, and 37 predictions) with 15 attributes (knowledge components) with error rates as values with some missing data. Every opportunity object was linked to its successor (next opportunity). The distances were calculated using the normalized Clark distance [3].

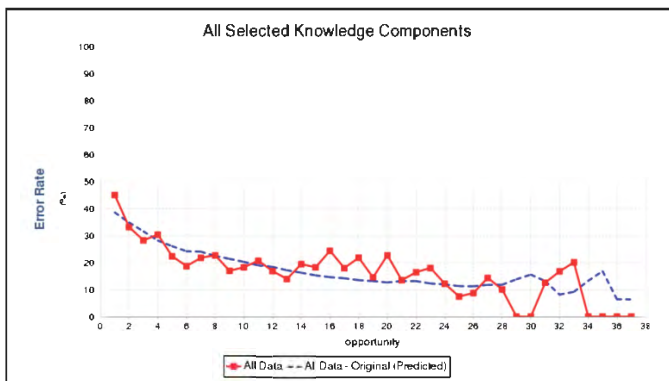


Figure 1. DataShop observed and predicted error rates for opportunities.

Figure 2 presents one view of the analysis. The 3D model shows a large dispersion of observation objects (red) as compared to the predicted values (blue). The specific identification of the object (not visible on the figure) coupled with the successor links indicate that the most dispersed observations are for the late opportunities.

Figure 3 shows the same 3D model but with a zoom near the collection of prediction objects. All the objects are in close proximity to their predecessor and successors, but for the lower right, which is the end of the opportunity sequence, capturing the sudden variability in the predictions as also visible from the DataShop plot (Figure 1). However, it is interesting to note that the learning curve pattern is not clearly visually identifiable from the 3D model. The main reason for this limitation is that each 3D model object captures the variation of its multiple attribute values (average error rates for each knowledge component), while the

learning curve in Figure 1 reduces this complexity to one measure of central tendency (average over all knowledge components).

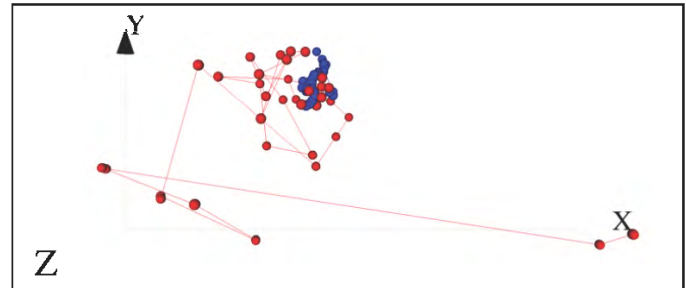


Figure 2. 3D opportunity objects observed (red) and predicted (blue) with error rates for KCs as attributes.

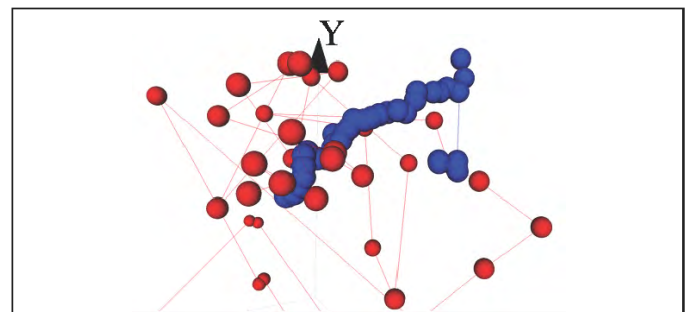


Figure 3. 3D opportunity objects observed (red) and predicted (blue) with error rates for KCs as attributes. (Zoom on predicted values).

2.2 Knowledge Components As Objects

The second analysis focuses on knowledge components as objects. This input data matrix contains 30 knowledge component objects (15 observations, and 15 predictions) with 37 attributes (opportunities). The matrix cells contained error rate values with some missing data. Every observed knowledge component object was linked to its predicted value.

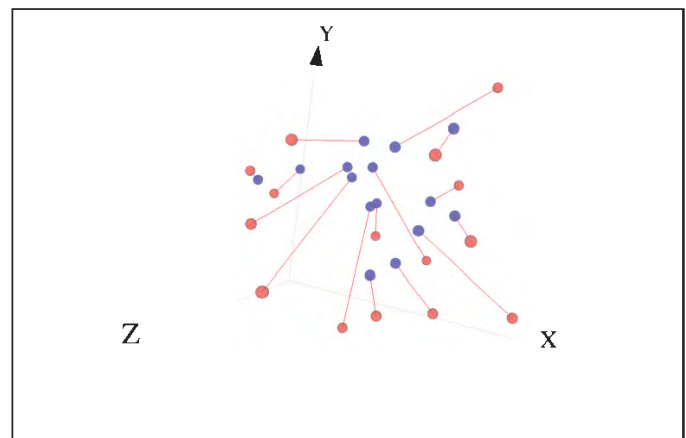


Figure 4. 3D knowledge component observations (red) and predictions (blue).

The Figure 4 presents the resulting 3D model. A visual inspection of the figure shows that observations (red) are more on the outside than the predictions (blue), suggesting that overall prediction values have more similarity than the observations. The 3D model also displays links of various lengths between knowledge component observations and predictions, suggesting a good fit for

short links. However, this 3D model does not match learning curve anomalies identified with specific knowledge components such as the COMPOSITION-BY-ADDITION in the original AFM model [8]. In the 3D model, the link between the observation and the prediction is not long (bad fit) and none of the nodes stand out of the main visual group of objects. Given that the data suffers from student attrition [6], it is possible that either missing values or high variability or error rates introduce too much noise and creates too much dispersion, so that some of the visual structure is not exposed.

In order to reduce the noise created by the attrition in late opportunities, a new input data matrix was generated by assigning opportunities to classes. Error rates values were split using a piecewise regression method in three opportunity phases: initial (opportunity 1), mastering (opportunities 2 to 7), and attrition (opportunities 8 to 37). This matrix contained 90 knowledge component objects which consisted of each of the 15 observed, and 15 predicted knowledge component error rate values for each of the 3 opportunity phases. Every observed knowledge component object was linked to its predicted value. Figure 5 presents a partial view of this analysis, leaving out the model the prediction objects. In the figure, the red objects correspond to the initial opportunity, the green object to the mastery learning efforts, and the blue objects to the remaining opportunities.

The figure indicates a cluster of red objects on the left and another one the right. The identification of these objects indicates that the objects on the left correspond to the knowledge components with high initial error rates. However, both the mastery (green) and attrition (blue) objects seem to be distributed evenly in the space. The figure also contains directional links going through some of the knowledge component objects for which there is at least a shift towards the left of the 3D model, indicating a learning curve anomaly.

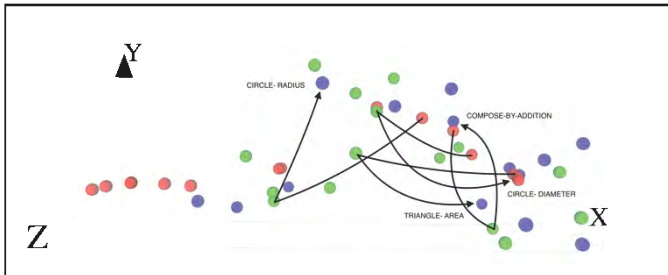


Figure 5. 3D knowledge component observations (KC) with initial opportunity (red), early mastery learning efforts (green), and late learning opportunities (blue). Along the X axis, it is expected that for a given KC, the order of colour from left to right would be red-green-blue.

Table 1 summarizes the main visual features of the knowledge components identified in Figure 5. A visual inspection of the figure indicates that the majority of objects (knowledge components) are ordered from left to right in red, green, and blue, which corresponds respectively to the classes of initial, mastery efforts, and late learning opportunities. However, some knowledge components in the figure break from this pattern, and are labelled with the KC names and arrows linking the learning stages. In agreement with other analysis of this dataset [7], the knowledge components COMPOSE-BY-ADDITION, CIRCLE-RADIUS, and TRIANGLE-AREA have been selected as learning curve anomalies and objects for model refinement [2]. Interestingly, the identification of potential issues was made possible by only looking

at the observations without having to inspect individual learning curves anomalies.

CONCLUSION

This poster/demonstration presented our preliminary efforts to explore learning data and model predictions by mapping them into objects in a three-dimensional space. Some limitations were obvious with the first analysis of the general learning curve over opportunities. However, the inspection of a knowledge component 3D model, where learning phases (initial, mastery, attrition) allowed to identify from only the distribution of the observations some knowledge components that could be targeted for model refinement. The approach seems to be valuable and worth pursuing for the combined analysis of learning data and model predictions. Future work will consist of looking at students' performance to see if the 3D model compares to AFM student parameter values. The approach could also be applied to visually compare alternative models of the same knowledge components set.

Table 1. Knowledge components with error rates change anomaly as identified in Figure 5.

Knowledge component	Initial to Mastery	Mastery to Attrition	Interpretation
COMPOSE-BY-ADDITION	Slight shift to the right	Slight shift to the left	No learning.
CIRCLE-RADIUS	Strong shift to the left	Slight shift to the right	Performance decrease then some learning
TRIANGLE-AREA	Strong shift to the left	Slight shift to the right	Performance decrease then some learning
CIRCLE-DIAMETER	Strong shift to the left	Strong shift to the right	Performance decrease then learning

3. REFERENCES

- [1] John R. Anderson. 2013. *Rules of the Mind*. Psychology Press. DOI:https://doi.org/10.4324/9781315806938
- [2] Hao Cen, Kenneth R. Koedinger, and Brian Junker. 2006. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. In *Intelligent Tutoring Systems. ITS 2006. Lecture Notes in Computer Science, vol 4053*, M. Ikeda, K.D. Ashley and TW. Chan (eds.). Springer, Berlin, Heidelberg, 164–175. DOI:https://doi.org/10.1007/11774303_17
- [3] Sung-Hyuk Cha. 2007. Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. *Int. J. Math. Model. METHODS Appl. Sci.* 4, 1 (2007), 300–307. Retrieved from http://users.uom.gr/~kouiruki/sung.pdf
- [4] Evanthia Dimara, Steven Franconeri, Catherine Plaisant, Anastasia Bezerianos, and Pierre Dragicevic. 2018. A Task-based Taxonomy of Cognitive Biases for Information Visualization. *IEEE Trans. Vis. Comput. Graph.* PP, 8 (2018), 1. DOI:https://doi.org/10.1109/TVCG.2018.2872577
- [5] A. Endert, W. Ribarsky, C. Turkay, B.L. William Wong, I. Nabney, I. Diaz Blanco, and F. Rossi. 2017. The State of the Art in Integrating Machine Learning into Visual

- Analytics. *Comput. Graph. Forum* 36, 8 (December 2017), 458–486. DOI:<https://doi.org/10.1111/cgf.13092>
- [6] Cyril Goutte, Guillaume Durand, and Serge Léger. 2018. On the learning curve attrition bias in additive factor modeling. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* 10948 LNAI, (2018), 109–113. DOI:https://doi.org/10.1007/978-3-319-93846-2_21
- [7] Kenneth R. Koedinger, R.S.J.d. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J Stamper. 2010. A Data Repository for the EDM community: The PSLC DataShop. In *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy and R.S.J.d. Baker (eds.). CRC Press, Boca Raton, FL.
- [8] Kenneth R. Koedinger, E. A. McLaughlin, and J. C. Stamper. 2012. Automated Student Model Improvement. In *Proceedings of the 5th International Conference on Educational Data Mining*, 17–24. Retrieved from http://educationaldatamining.org/EDM2012/uploads/procs/Full_Papers/edm2012_full_10.pdf
- [9] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE Laurens. *J. Mach. Learn. Res.* 9, (2008), 2579–2605. DOI:<https://doi.org/10.1007/s10479-011-0841-3>
- [10] Brent Martin, Antonija Mitrovic, Kenneth R. Koedinger, and Santosh Mathan. 2011. Evaluating and improving adaptive educational systems with learning curves. *User Model. User-adapt. Interact.* 21, 3 (August 2011), 249–283. DOI:<https://doi.org/10.1007/s11257-010-9084-2>
- [11] Allen Newell and Paul S. Rosenbloom. 1981. Mechanisms of Skill Acquisition and the Law of Practice. In *Cognitive Skills and Their Acquisition*, John R. Anderson (ed.). Routledge, Taylor & Francis Group, New York, NY, 1–56. DOI:<https://doi.org/10.4324/9780203728178>
- [12] Peter Pirolli and Stuart Card. 2005. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis*, 6.
- [13] William Ribarsky and Brian Fisher. 2016. The Human-Computer System: Towards an Operational Model for Problem Solving. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, 1446–1455. DOI:<https://doi.org/10.1109/HICSS.2016.183>
- [14] J.W. Sammon. 1969. A Nonlinear Mapping for Data Structure Analysis. *IEEE Trans. Comput.* C–18, 5 (May 1969), 401–409. DOI:<https://doi.org/10.1109/T-C.1969.222678>
- [15] Julio J. Valdes. 2017. Visual Data Mining with Virtual Reality Spaces. In *Big Data and Analytics Applications in Government*. Auerbach Publications, 131–158. DOI:<https://doi.org/10.4324/9781315153582-7>
- [16] Julio J. Valdés. Virtual Reality Representation of Information Systems and Decision Rules: An Exploratory Technique for Understanding Data and Knowledge Structure. In *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*. Springer, Berlin, Heidelberg, 615–618. DOI:https://doi.org/10.1007/3-540-39205-X_101

A generalizable performance evaluation model of driving games via risk-weighted trajectories

Rory Flemming
University of Minnesota
flemm053@umn.edu

Emmanuel Schmück
University of Luxembourg
emmanuel.schmuck@uni.lu

Dominic Mussack
University of Luxembourg
dominic.mussack@uni.lu

Pedro Cardoso-Leite
University of Luxembourg
pedro.cardosoleite@uni.lu

Paul Schrater
University of Minnesota
schrater@umn.edu

ABSTRACT

Efficient learning experiences require content to dynamically match a learner's skill; this assumes a fast and accurate assessment of the learner's skill and the ability to update content accordingly. Effective personalized learning therefore involves deriving a performance-predictive mapping between behavioral and environmental factors. Once learned, this relationship can be used to generate new content and to update skill estimates based on the learner's interactions in an adaptive system. To provide proof of concept:

(1) We develop a fast-paced driving video game where the player skillfully navigates a cluttered environment comprising obstacles and collectibles. Game content is generated procedurally and player behavior is recorded in the game—this provides an ideal test-bed for a method aiming to learn such a performance-predictive mapping.

(2) Using blurred occupancy maps of the game's segments, we generate risk-weighted trajectory profiles for each user and segment of the game. Here, we show that these profiles can be used in a regression model to predict in-game performance both within and between game segments. Additionally, these profiles themselves reveal a trade-off between in-game rewards and risks. Successful identification of predictive environmental units within the game provides insight into the mapping between environmental features and performance, while facilitating the process of procedurally generating new, appropriate content in our adaptive system. We show that rapidly assessed measures of risk are highly predictive of both driving performance and reward rate, providing proof-of-concept evidence for the feasibility of a personalized adaptive learning system for this game.

Keywords

Game-based learning; serious games; dynamic difficulty adjustment; regression; interactive learning environments

1. INTRODUCTION

A central topic in modern educational research is to determine how best to personalize learning experiences. While mismatched content can lead to inefficient learning, boredom, frustration, and disengagement, well adapted content may lead to Flow state, and learning efficiency [1]. Personalized learning systems require a) a *performance evaluation mechanism*—in order to measure the impact of content features on learner performance—and b) an *adjustment mechanism*—that determines content updates depending on the performance evaluation [2]. Hence, personalized learning systems require the derivation of a performance-predictive mapping between learner's performance and content features. Such a mapping is particularly challenging to derive for complex environments with continuous, real-time user interactions.

We aim to develop a framework to investigate solutions for real-time performance evaluation and content adjustment in a complex environment. Such a framework requires the ability to generate content procedurally with complete control over the content features. It must also be able to provide continuous interactions between content and user, support the data collection related to these interactions, and must be accessible and engaging to a wide audience. We use a driving video game as the learning environment because it fulfills all of these requirements. In addition, driving requires motor control, which is a well characterized domain for understanding general principles of human learning on a computational and biological basis, [3] and it has been used for cognitive training [4]. A driving game might seem unrelated to traditional educational settings (e.g., math quiz); yet, we believe that it provides a fertile ground for research on personalized learning and has the potential to transfer to situations that are more typical in education.

To make the analogy to education clearer, we treat the procedurally generated driving tracks as being composed of a sequence of segments, each of which presents a navigation problem. Segments tap navigation knowledge, and we assume there is a low-dimensional knowledge space that covaries with

Rory Flemming, Emmanuel Schmück, Dominic Mussack, Pedro Cardoso-Leite and Paul Schrater "A generalizable performance evaluation model of driving games via risk-weighted trajectories" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 548 - 551

measurable features of the segments, and performance on these segments. Our goal is to show how to use player performance to the parse the environment into segments predictive of differences in performance, then use the features of these segments to learn which aspects of the procedural generation system most affect segment “difficulty”. The end result is an online segment-wise performance evaluation mechanism that uses segment feature information to predict difficulty, and uses observed performance to estimate skill.

In complex learning environments, like in this driving game, quantifying difficulty needs to integrate performance with respect to all task objectives; because players must simultaneously reduce the risk of crashing while maximizing collectibles, we model difficulty as having risk and reward components. In general, tasks with multiple objectives should manifest trade-offs, which makes simple performance assessments inappropriate. Thus, simultaneous assessment of risk scores and reward rates are essential for investigating the validity of an adaptive system framework that we wish to implement.

1.1 Adaptive System

The flow of the proposed online adaptive system is shown in Figure 1. In the learning environment (here, the driving game), content is procedurally generated, varying key parameters which affect difficulty. This new content is processed to identify performance predictive features of the content by creating a risk prediction map. This map can be used to predict user performance on content and identify ahead of time which sections of the map are likely to be predictive of success (e.g., acquiring collectibles, avoiding collision). After the user interacts with the content, feedback about risk (e.g., proximity to obstacles) and reward (e.g., observed collection rate) is used to update the user’s skill estimate in the trade-off space. In this paper, we focus on *proof of concept* in generating, within the context of driving video game, the risk prediction map and validating its relationship with performance on our segments.

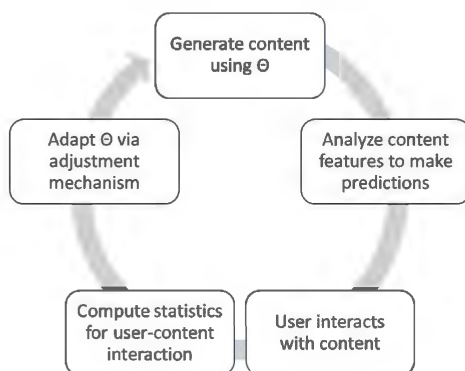


Figure 1. Flowchart of the full adaptive system. Game content is generated by specifying parameters for a generative process. These features have a relationship to the content’s difficulty, and when learned, provide a prediction of performance. After presentation, interaction statistics are calculated and the estimate of the individual’s skill is updated. The skill estimate is used to set new environment parameters, Θ . To learn the relationship between content features and performance, the intermediary relationship between interaction statistics and both performance and content features must be observed.

2. METHODS

2.1 Game Design

We developed a fast-paced driving video game where the player has to skillfully navigate a vehicle through a cluttered environment comprising various obstacles and collectibles (see Figure 2). The vehicle moves forward at a constant speed and the player controls its translational movement by pressing the right or left arrow keys in order to acquire collectibles (by running into them) and to avoid the obstacles which delimit a path to follow. Colliding with an obstacle causes the vehicle to reset at an earlier point on the path with no speed; speed then gradually ramps up over the course of 3 seconds before reaching its constant value. During gameplay, the player’s inputs, trajectory, collectibles collected and collisions are recorded.

The content of the game (i.e., the path to follow, the distribution of obstacles and collectibles, and the visual aesthetics) is generated procedurally. The driving space is composed of a sequence of independent segments, separated by buffer zones which contain no obstacle or collectible. Each segment is generated from a procedural sequence of user inputs describing a timeline of keys (left, right) being pressed and released, for a given duration—this ensures that in principle the segment can be successfully driven. This sequence of virtual user inputs can be characterized by its *variability* (i.e., change in duration and direction across inputs), and its *pacing* (i.e., number of inputs over time) and is used to compute the corresponding *reference trajectory* that a player should follow, taking into account the physical characteristics of the vehicle such as mass, drag and thrust. Obstacles are then placed around this reference trajectory. The minimal distance between obstacles and trajectory defines the *narrowness* of the segment. The minimal distance between obstacles defines the *ambiguity* of the path—when the distance is small the correct path is perceptually salient and there is low ambiguity as to where to drive; when the distance is big it becomes unclear which path to take (see Figure 2). Finally, collectibles are placed directly on the reference trajectory, at critical points of each turns.

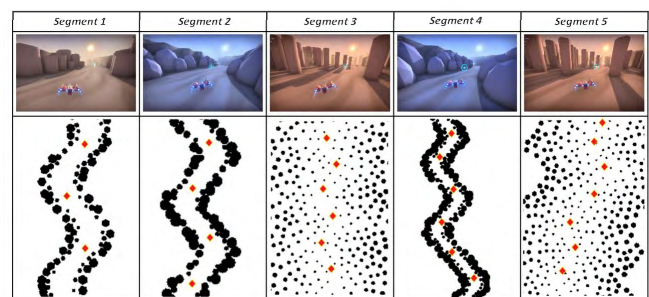


Figure 2. Subjective (top) and bird-eye (bottom) view for representative portions of each segment (obstacles in black, collectibles in orange). Each segment represents a distinct statistical profile of our generative game design. A video of the game can be found here: https://youtu.be/gRSX_cOh5-4

2.2 Reward Rate Scores

To quantify a player’s performance on the task of acquiring collectibles for a given segment we use the reward rate score which is defined as:

$$\lambda = (c/t) / (C/T)$$

Where C and T are parameters of the segment and c and t are behavioral data. Specifically, C is the total number of collectibles on a segment and T the fastest possible completion time of that segment; c is the number of collectibles acquired by the player and t is the players' actual completion time. Therefore, λ is a proportion between the rate at which the collectibles were acquired (i.e., c/t) and the highest possible rate for that segment (i.e., C/T).

2.3 Risk Scores

We postulate that an individual's ability to achieve a high reward rate score (λ) is a function of their ability to acquire collectibles while avoiding obstacles. To quantify the individual's interactions with the environment, we compute a risk statistic for their trajectory. This statistic is a function of both features of the environment and the user's behavior. We compute a risk score for each observed player trajectory through the segment, which weights the trajectory by its distance to nearby obstacles at each time point, monotonically resembling the risk of collision. Since collectibles are placed along the reference trajectory used to generate the segment, they reside along points where this function will be low, while space occupied by obstacles and near obstacles is high.

A vectorized bird's eye map of each segment (see Figure 2, bottom) can be used to generate a risk profile for each measured trajectory. Since risk of collision should drop off continuously with the distance between the vehicle and obstacles, a gaussian kernel was convolved with a filtered map containing only obstacles to generate the risk heat map. Then, for each point in a player's trajectory, the corresponding risk value on the map is found in order to construct a risk profile for that trajectory. That is, the risk value of every point in the trajectory. Averaging the risk values across the trajectory provides a unique *Risk Score*.

2.4 Pilot Study Design

For this pilot study, we generated one track that was composed of five different segments; the exact same track was presented to all participants. We generated segments that allowed us to span the space of relevant generative parameters and provided enough samples in terms of observed trajectory length. The first segment is a succession of identical, wide, long turns with no gap between obstacles. Subsequent segments are designed to explore various combinations of features (i.e., different values of narrowness, pacing, ambiguity, variability) and are expected to be of increasing difficulty, culminating with the last segment where perfect performance is only achievable for an expert player. Table 1 illustrates the parameter choices for these five segments.

Table 1. Qualitative descriptions of each segment.

	segment 1	segment 2	segment 3	segment 4	segment 5
Narrowness	low	medium	medium	high	high
Pacing	low	medium	medium	high	high
Variability	low	low	medium	high	high
Ambiguity	low	low	high	low	high

2.5 Data Set

Seven participants played the driving game one to seven times, resulting in 28 sessions with a unique trajectory for each of the five segments; it takes a minimum of 26 seconds to complete a single segment and the longest observed completion time was 138 seconds (on segment 5). The main role for this preliminary dataset is to serve as a testbed for exploring the relationship between the risk scores of a trajectory and the individual's performance in environments generated procedurally from a set of parameters. This understanding contributes to the development of an analysis pipeline whereby these relationships could, in a future step, be exploited by an adaptive difficulty algorithm.

2.6 Analysis

To assess the relationship between risk and reward scores, we looked at a) the relationship between a single segment's risk score and performance and b) the relationship among segments to see whether information from one segment generalizes to other segments. Normally, a regression model predicting a rate would use a Poisson link, however, since the data collected fell mostly in the middle (linear) region of possible values, a log-linear model was used instead, with the log transforming the data to better fit normality assumptions.

Cross-segment predictions could be informative for an adaptive system which aims to predict performance on one segment from the performance on a previous one. The risk scores of each of the 5 segments for each of the 28 sessions were compiled into a matrix (28 sessions x 5 segments) for fitting linear regression models for each segment where the output variable is the log-transformed reward rate score (λ) for each attempt on the segment. We fit the relationship between the risk scores one or multiple segments and the reward rate score of each segment, giving five models with five possible predictors each. Additionally, we look at the specific within-segment cases.

The model search space for cross-segment fits is defined by all combinations of the main effects of risk score for each segment ($2^5 = 32$ possible models per segment). The best-fit model of the reward rate score on each segment was selected using an exhaustive branch-and-bound search algorithm with Mallows' Cp criterion [5]. In the case of gaussian linear regression, Cp values are equivalent to AIC. This procedure is functionally similar to lasso regression by forcing some parameter coefficients to be 0 by not including them when they do not contribute to the fit. Thus, five within-segment relationships and five between-segment models are considered.

3. RESULTS

Linear fits of the relationship between reward rate on a segment and the mean risk score for that segment is shown in Figure 3. We see that overall, the reward-rate is negatively associated with the risk score, characterizing a trade-off. Additionally, this tradeoff varies between segments, with some segments producing different ranges of both reward-based performance and risk-based performance. The coefficients of the risk scores of one segment in the best fit model of the reward rate of another segment is shown in Figure 4. Notably, the risk scores of segments show stronger relationships with the reward rates of segments with more similar features and difficulty (Table 1).



Figure 3. Difficulty is captured through risk-reward trade-off functions. The relationship between the mean risk score on a segment and the log of the normalized reward rate. Each datapoint represents one trajectory through a segment. The slope shows a trade-off between risky behavior and reward-rate, such that incurring additional risk negatively affects the rate at which collectibles are acquired. The slopes present a distinctive notion of difficulty—higher slopes are more challenging in the sense that there is less leeway for trajectory error to preserve performance. Note that the *average performance* per segment does not capture this more fundamental trade-off. In an adaptive system, we target the relationship between segment attribute statistics and the slopes and intercepts of the risk-reward trade-off functions.

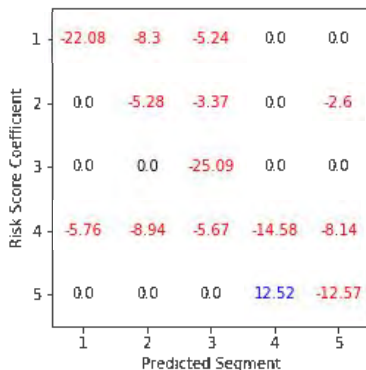


Figure 4. Difficulty varies smoothly with task statistics. Matrix showing the coefficients of risk scores in best fit models for cross-segment predictions of reward rates. Each ij -cell represents regression coefficient for predicting reward success on segment j from the risk profile in segment i .

4. DISCUSSION AND FUTURE WORK

Our results demonstrate a fundamental trade-off in difficulty in our driving game, and that difficulty is directly related to the underlying task generative parameters. Moreover, segment-wise performance is predictive of performance on new segments, indicating that there is some lower-dimensional space that indicates skill *across* racing segments. The simple linear models (shown in Figure 3) and best-fit linear models (shown in Figure 4) suggest that in our virtual environment the analysis of the risk profiles of player trajectories on some segments is informative of a reward rate-based performance measure on the same and other segments (R^2 range from 0.58 to 0.74). While measuring reward

rate is slow during online gameplay, measuring risk is fast and relatively easy. This analysis demonstrates we can predict both using only risk scores, which is important during real-time assessment.

Given our ability to diagnose performance, we can incorporate our results into a full adaptive system by using the predicted skill to adjust difficulty. This can be incorporated directly, for instance by having a library of segments that are adaptively selected based on their predicted difficulty for a given user. To incorporate the procedural generation aspect of the game we can sample the generative parameter space to predict this difficulty tradeoff in a hierarchical model. This can close the loop on the adaptive system in Figure 1, by adjusting segment generation based on risk assessment.

5. CONCLUSION

We proposed a framework which makes use of the relationship between features of procedurally generated content and individual performance to generate novel content. Using a modest sample of pilot data collected from a custom-designed driving game, we show that a risk statistic adequately summarizes the interactions between a player and segments of the game. The relationship between these variables reveals a tradeoff between performance and risk in the task. Using this statistic, which can be computed easily online, predictions about performance both on current task and future tasks can be made. In generating an operating adaptive system, the next step will be to learn a model of the relationship between environmental features and these summary statistics to facilitate the adaptive generation of content.

6. ACKNOWLEDGMENTS

This research was supported by the Luxembourg National Research Fund (ATTRACT/2016/ID/11242114/DIGILEARN) and (INTER Mobility/2017-2/ID/11765868/ULALA).

7. REFERENCES

- [1] Craig, S., Graesser, A., Sullins, J., & Gholson, B. 2004. Affect and learning: an exploratory look into the role of affect in learning with AutoTutor. *Journal of educational media*, 29(3), 241-250. DOI: <https://doi.org/10.1080/1358165042000283101>
- [2] Streicher A., Smeddinck J. 2016. Personalized and Adaptive Serious Games. In Dörner, R., et al. ed. *Entertainment and Computing and Serious Games*, Springer, Cham, 2015, 332-377.
- [3] Shadmehr, R., & Mussa-Ivaldi, S. 2012. *Biological learning and control: how the brain builds representations, predicts events, and makes decisions*. Mit Press, Cambridge, MA.
- [4] Forman, G. 2003. *J. Mach. Learn. Res.* 3 (Mar. 2003), 1289-1305.
- [5] Anguera, J. A., Boccanfuso, J., Rintoul, J. L., et al. 2013. Video game training enhances cognitive control in older adults. *Nature*, 501(7465), 97-101. DOI: <https://doi.org/10.1038/nature12486>
- [6] Lumley, T. 2017. leaps: Regression Subset Selection. R package version 3.0. Based on Fortran code by Alan Miller. <https://CRAN.R-project.org/package=leaps>

Visualization and clustering of learner pathways in an interactive online learning environment

Daniel Furr
Pearson
50 California St.
San Francisco, CA 94111
daniel.furr@pearson.com

ABSTRACT

The present study seeks to improve our understanding of how learners interact with assignments in an interactive online learning platform, Revel. First, a means of visualizing learner pathways through assignments is proposed. This tool is then used both to inform choices regarding feature extraction and to interpret the result of applying cluster analysis to those features. Results suggest that characteristic ways of interacting with the online platform include: completing assignments while largely following the designed sequence of activities, completing assignments while deviating more from the sequence, and stopping work before completion of the assignment.

Keywords

interactive learning environment, visualization, cluster analysis, partitioning around medoids, k -medioids

1. INTRODUCTION

Traditionally, active engagement in higher education courses requires learners to spend time reading sections of an assigned textbook. Reading material can reinforce what is taught in class, but learners may be discouraged from reading when the text is difficult to access and is not engaging. The present study explores learner behaviors in an interactive online learning platform, Revel. With Revel, learners are given assignments to be completed before class so that they can arrive to class better prepared to learn. Typically, an assignment covers one chapter, broken into sections. These assignments alternate between learning objects (texts, videos, interactives, etc.) and assessment objects (writing prompts, multiple choice questions, etc.). The assessment activities help the learners to check their understanding. At the end of a chapter, the learners are presented with a more summative assessment of the content.

Though learners within a class will typically see the same online assignments, they likely will not engage with the assignments in the same way. The present study seeks to improve our understanding of how learners may differentially interact with online assignments.

First, a means of visualizing learner pathways through assignments is proposed. This tool is then used both to inform choices regarding feature extraction and to interpret the result of applying cluster analysis to those features. In the end this approach should indicate several characteristic ways in which learners work through the online assignments.

2. SAMPLE

Six introductory psychology classes were taught at the same university. In each class, thirteen assignments were administered with the online platform, and these assignments were similar but not identical between classes. In total there were 761 learners who altogether submitted 8,341 assignments in the online platform.

3. METHODS

3.1 Data

The stream of learner interactions within the online platform was extracted from the platform database. Each interaction may be characterized by the type of object involved and the type of action performed.

The three object types are:

- *learning objects*: the texts, videos, interactives, etc. that make up the instructional aspects of the chapter
- *section assessment objects*: questions posed to the learner following a small set of learning objects
- *chapter assessment objects*: summative questions posed at the end of the assignment

The two actions types are:

- *view*: the learner accesses a learning object or accesses an assessment object without answering it
- *answer*: the learner provides an answer to an assessment object

The assessment objects are multiple choice questions, which may be attempted up to three times. Three points are awarded for correct responses to section assessment objects, while five are awarded for chapter assessment objects. A one-point penalty is applied to reattempts. Instructors are encouraged to incorporate assignment scores into the course grade to bolster learner engagement.

3.2 Visualization of pathways

A learner's pathway through one assignment may be depicted visually. Figure 1 provides an example. In the figure, the actions a learner takes are displayed as connected points, arranged along the

Daniel Furr "Visualization and clustering of learner pathways in an interactive online learning environment" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 552 - 555

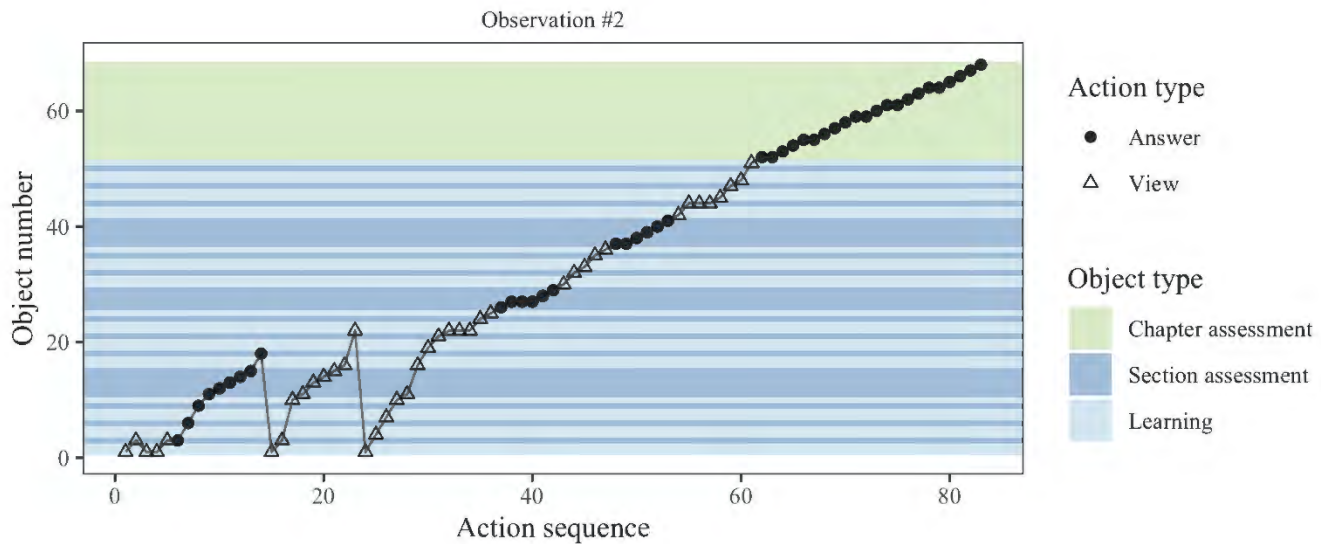


Figure 1. Example pathway.

x-axis in the ordered that the learner took them. The shape of the points indicates the type of action: circles for answering and triangles for viewing. The sequence of objects for the assignment is given in the y-axis, and the colored horizontal ribbons divide the plot region according to object type: lighter blue for learning objects, darker blue for section assessment objects, and green for chapter assessment objects.

The example pathway in Figure 1 shows that this learner started by accessing a couple of learning objects two to three times each. Then the learner answered nine section assessment objects in sequence, skipping over several learning objects as they did so. Following that, the learner backtracked to review many of the objects they previously encountered without providing new answers to the assessment objects. Next, the learner backtracked again, viewing some of the learning objects they previously skipped before continuing on through the assignment in sequence.

3.3 Feature Extraction

Reviewing several example pathways as in Figure 1 informed on what features should be extracted from the stream of interactions. Based on this review, the following features were extracted for the 8,341 assignments submissions:

- *sections_viewed*: the percentage of learning objects the learner viewed
- *sections_repeated*: the percentage of learning objects the learner viewed more than once
- *section_items_answered*: the percentage of section assessment objects the learner answered
- *section_items_repeated*: the percentage of section assessment objects the learner answered more than once
- *chapter_items_answered*: the percentage of chapter assessment objects the learner answered
- *chapter_items_repeated*: the percentage of chapter assessment objects the learner answered more than once
- *skip_aheads*: the percentage of transitions between objects in which the learner moved forward in the sequence skipping over one or more objects
- *go_backs*: the percentage of transitions in which the learner went to an earlier object in the sequence

3.4 Clustering of pathways

Cluster analysis is performed to determine what typologies may exist for student pathways through the online assignments. The Partitioning Around Medoids [1] (PAM) algorithm, also known as *k*-medoids clustering, is employed for this purpose. PAM is similar to *k*-means clustering [2], but rather than define clusters by means, it defines clusters by their most representative observation (mediod). An advantage of PAM is that it is more robust to outliers than *k*-means clustering.

For both PAM and *k*-means clustering, it is necessary to prespecify the number of clusters *k*. Clustering methods are commonly used in exploratory analysis where *k* is not known in advance. The choice of *k* may be made by trying several values and selecting the one that maximizes some criteria. The criteria used in the present study is the average silhouette width [3], which provides a measure of how well each observation fits with its assigned group, ranging from zero to one. Averaging this across the dataset provides an overall index of the goodness of fit that may be used to select an optimal *k*.

4. RESULTS

The PAM algorithm was applied for values of *k* ranging from 1 to 7 on standardized (mean-centered and divided by the standard deviation) features, using Euclidean distance as the dissimilarity measure. The average silhouette width was maximized at .24 for *k* = 3 groups. This value for average silhouette width is considered low and suggests a somewhat weak structure regarding the clusters.

Focus will be given to the three-group solution because the greatest support was observed for it. The groups, numbered one through three, consisted of 2,852, 4,832, and 657 observations respectively. Figure 2 presents the (unstandardized) feature values for the representative observations of the groups.

Looking to Figure 2, Group 1 may be characterized as viewing about half of the learning objects, answering about half of the section assessment objects, and answering all or nearly all of the chapter assessment objects. This group also tended to view learning objects more than once, and often skipped ahead or went back in the sequence of objects. For convenience of interpretation, this

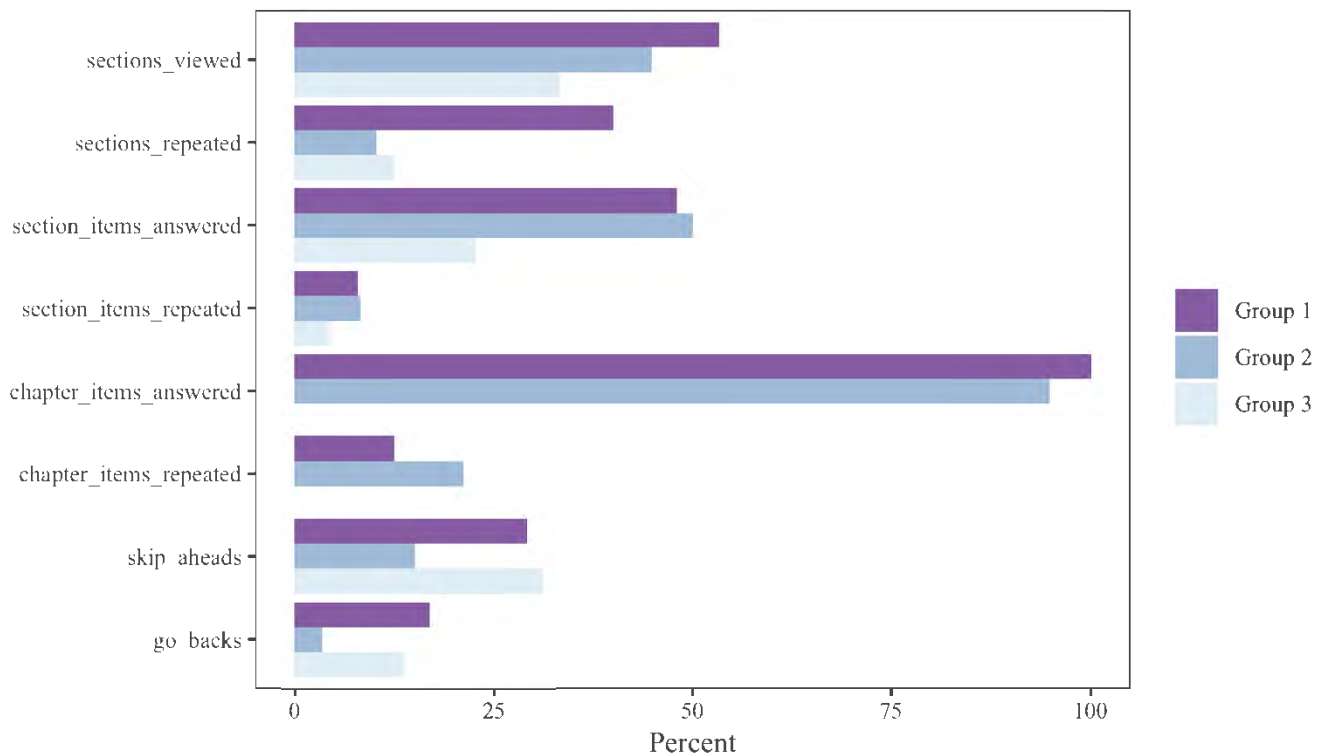


Figure 2. Feature values for the representative observations.

group may be labeled *wanderers*, as they engaged with most of the objects but not in an orderly sequence.

Group 2 is similar to Group 1 in terms of the number of objects viewed and answered, but Group 2 less often skipped ahead or went back in the sequence. Their pathway through the assignment was more linear, and for this reason Group 2 may be labeled *arrows*.

Group 3 is characterized as viewing fewer learning objects, answering fewer section assessment objects, and answering few or no chapter assessment objects. Given that the chapter assessment objects appear at the end of the assignments, these points suggest that this group quits assignments part way through. In light of this, Group 3 may be labeled *incompleters*.

A feature of the PAM algorithm is that each cluster is formed on the basis of a representative observation. Similar to the visualization for the example pathway shown earlier in Figure 1, it is possible to visualize the three representative observations as shown in Figure 3. (Note that the three observations are not associated with the same assignment and so have different numbers of objects depicted in the figure.) Naturally, the subplots in Figure 3 mimic the preceding characterizations of the groups; the wanderer proceeded through the assignment in a more erratic path than the arrow, and the incompleter stopped part way through.

5. DISCUSSION

In the present study, a means of visualizing learner pathways through the online assignments was developed and subsequently used to guide in feature selection and interpretation for a cluster analysis. The results of the analysis suggest that learner pathways may be characterized as wanderers, arrows, or incompleters, with these group memberships encapsulating how much and how orderly assignments were completed. Future research may

investigate the relationships between group membership and outcomes such as course performance.

The result of a cluster analysis will naturally be a function of the features included, and so it must be considered that if different features were extracted for these data, then the clusters revealed would also differ. Further, given that the value for average silhouette width for the selected clustering scheme indicated weak structure, it is likely that an improved set of features could be devised. A better feature set could include information about the correctness of answers to assessment objects, the amount of time spent on the different types of objects, or when the work is done in relation to the due date. Future research may investigate these additions.

6. REFERENCES

- [1] Kaufman, L., and Rousseeuw, P. J. 1987. Clustering by means of medoids. In *Statistical Data Analysis Based on the L-Norm and Related Methods*, Y. Dodge, Ed. North-Holland, 405–416.
- [2] MacQueen, J. B. .1967. Some Methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability* (Berkeley, CA, June 21, 1965). University of California Press, Berkeley, CA 281–297.
- [3] Rousseeuw, P. J. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Computational and Applied Mathematics*. 20 (Nov. 1987), 53–65.

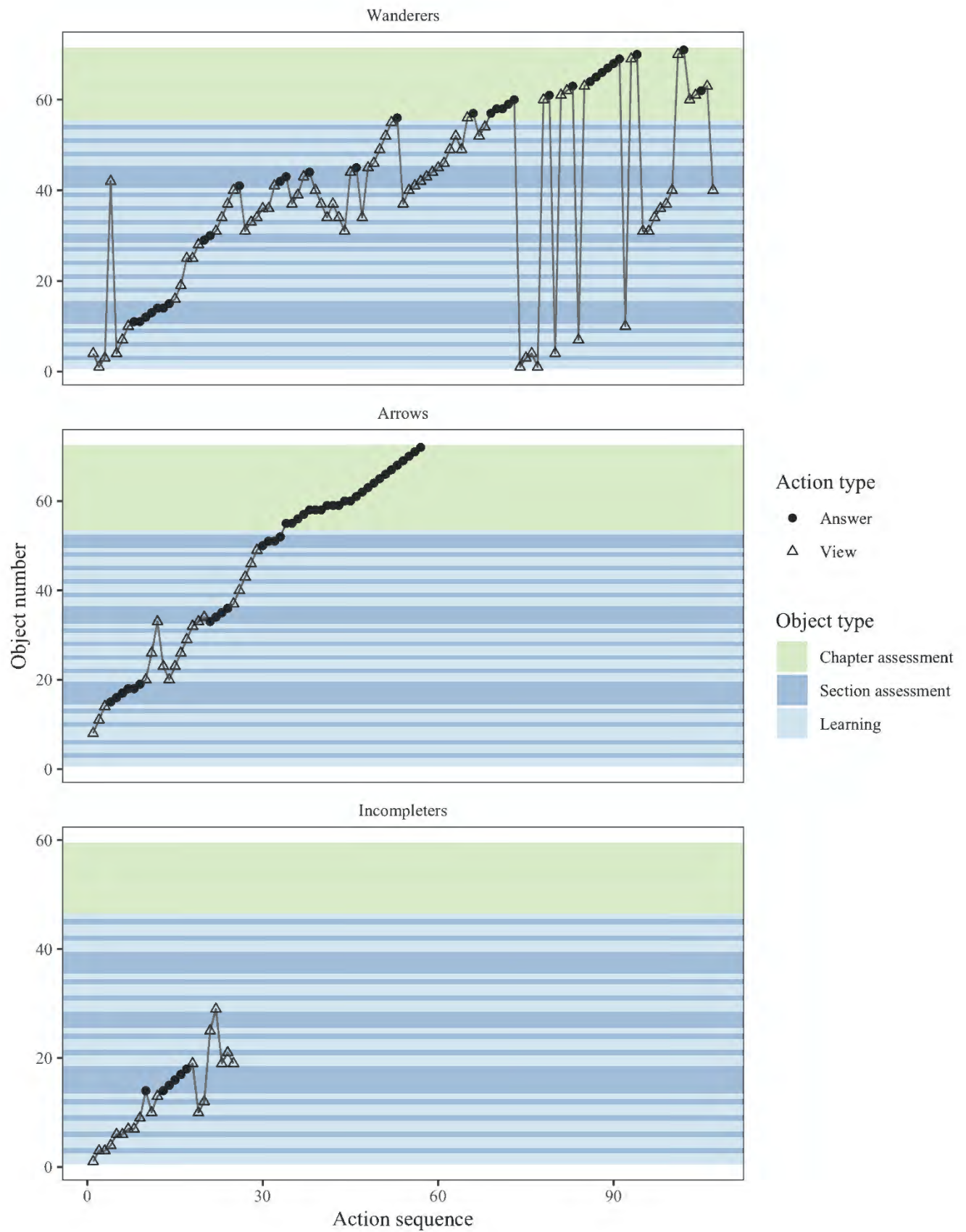


Figure 3. Representative pathways for the three groups.

Filtering non-relevant short answers in peer learning applications

Vincent Gagnon
Polytechnique Montréal

Audrey Labrie
Polytechnique Montréal

Sameer Bhatnagar
Polytechnique Montréal

Michel C. Desmarais
Polytechnique Montréal

ABSTRACT

Applications that adopt peer instruction and active learning make use of short student answers as a learning opportunity: they take some answers as topics of discussion in class, or select answer rationales from peers to foster self-reflection on the learner's own rationale to a question. However, experience with these applications reveals that some answers are irrelevant, inappropriate, or could even be offensive, and must therefore be filtered out. Automatic identification of these rationales is the topic of this research. We introduce an easy-to-implement approach based on standard text classification techniques, namely bag of words and vector space models, and show its effectiveness for filtering irrelevant answers.

Keywords

Student response systems, Short answer grading, Vector space model

1. INTRODUCTION

Contemporary student response systems (SRS) [4, 2] allow students to provide free-text answers to short open-ended questions. These open-ended questions better enable teachers to capture student's higher level of understanding than close-ended questions [8]. Answers to open-ended questions can serve for in-class discussion and feedback, such as in Socrative [6]. They also serve in out-of-class peer-instruction, as in DALITE [1] where other student's answer rationales are presented to students to nurture self-reflexion on a student's own rationale.

However, experience with SRS shows that a small proportion of the answers given by students are inappropriate and should not be presented to their peers. Filtering out these answers by reading through them is a daunting task. For example, with over 100k answer rationales currently in our

own system, DALITE, the task is highly demanding. With answers that are shown in-class, as with Socrative, pre-screening would take up too much teacher attention and would not scale well to large classes. An automatic means to filter unwanted answers is therefore paramount to such SRS. The goal is to identify student answers that are deemed irrelevant or inadequate for in-class discussion or for inducing student self-reflection on the topic.

This problem is similar to spam-filtering to the extent that it is a binary classification to filter undesired answers, but it also draws from the problem of automatic short answer grading to the extent that desired answers are related to a topic and to a specific question. While the topic of automatic short answer grading dates back to the 1960's and is still an active field of research (see [3, 5, 10] for some recent advances), we propose an approach to filter unwanted student answers that relies on a simple Bag of Words (BOW) method to classify undesired answers from relevant ones.

2. PROPOSED FILTERING METHOD

Building on the idea that a binary relevant/non-relevant classification is the goal and on the principle that correct answers will dwell on a specific topic, the proposed method rests on the principle that irrelevant student answers will contain words that are unrelated to the ones of the relevant answers. In that respect, an irrelevant answer is expected to be close to orthogonal to all other answers in a vector space model [11].

Examples of non-relevant answers are, for eg.:

Books for sale
idk (for "I do not know")
I think thats what it is because i am guessing
Who knows, man. It's gotta be constant
four more years!
.
a (for choice a)

Vincent Gagnon, Audrey Labrie, Michel Desmarais and Sameer Bhatnagar "Filtering non-relevant short answers in peer learning applications" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 556 - 559

As can be seen, answers are often very short, but not always. They are sometimes humorous or sarcastic, or simply expressions of frustrations or disengagement. But they are characterized by a vocabulary that is unrelated with the questions domain: physics and ergonomics in our two corpora. Yet, the vocabulary used is generally non-specific to

the domain of the question, which is the assumption behind the proposed approach.

To measure the general orthogonality of an answer, we average the cosine of each answer with all other answers. Given a document-term matrix of n student answers (documents) by t terms, $\mathbf{A}_{n \times t}$, the average cosine of answers d_i in corpus D is defined as:

$$\overline{\cos}_D(d_i) = \frac{\sum_{j \in D} |\cos(d_i, d_j)|}{|D|}$$

where $|D|$ is the corpus size and $|\cos(d_i, d_j)|$ is the absolute value of the cosine between answers d_i and d_j .

To create the term-document matrix, $\mathbf{A}_{n \times t}$, we apply the common practice of word stemming and spell correction. The `tm`¹ package is used for this purpose [7]. The term-document matrix is also transformed using the standard TF-IDF calculated from the corpus.

2.1 Answer relevance classification

The next step is to use the average cosine, $\overline{\cos}_D(d_i)$, to determine if an answer is relevant or not.

We can see from Figure 2 that the distribution of the average cosine for relevant and non-relevant answers is bimodal. Figure 2's distribution corresponds to a single corpus, but both corpuses show the same bimodal pattern. This observation leads to a simple approach to determine the non-relevant answers based on clustering.

Given that non-relevant answers are a small portion of all answers and their average cosine is always lower than the large majority, they tend to cluster around and above 0. We explored the k-means clustering algorithm on the average cosine dimension and find that for values between 2 and 10 clusters, the lowest cluster leads to a relatively stable set of answers. Therefore, the cluster with the closest value to 0 is considered as the non-relevant set of answers. For this study we chose the value of 4 for the number of clusters.

3. DATA SETS

Two corpuses are used in this study. They cover two contexts of use and, to a certain extent, two extremes cases: a small data set with multiple valid open text answers, and a larger data set of rationales to a specific answer choice in a multiple-choice question.

The first corpus is small and representative of the context where answers must be filtered in real-time as they come in. The second corpus is from a peer learning environment where filtering can be conducted off-line. The objective of this environment is to avoid presenting self-reflective rationales that are non-relevant. The first corpus is typical of *Socrative* whereas the second one is from *DALITE*. Another factor that differentiates the two corpuses is students were anonymous in the first context whereas they were not in *DALITE*.



Figure 1: Corpus 1: usability issues with an MP3-CD player device.

Table 1: Inter-judge agreements

	Corpus 1	Corpus 2
Number of non-relevant texts found by Judge 1	11	286
Number of non-relevant texts found by Judge 2	10	297
Inter-judge agreement (Kappa)	95.0 %	97.0 %

Corpus 1. The first corpus is a small set of answers to a single question: “What are the usability issues with the CD-player device”. The device is shown in Figure 1 and has a fair number of problems over many dimensions, from *guidance* [9] to requirements engineering. The question was proposed to a class of approximately 150 students and the students answered through the *Socrative* application. Since there are multiple good answers, the answers often reported multiple issues with the devices. Answers were segmented into single-issue texts by two teacher assistants. Inter-judge reliability was computed using Cohen’s Kappa statistic for two judges who classified answers as relevant or not. As shown in Table 1, the inter-judge agreements are nearly perfect.

Corpus 2. The second corpus contains a much larger number of student answers and covers eight questions. Segmentation of answers into issues addressed is not relevant in this context. The answers are taken from a college-level physics class.

General statistics on the two corpuses are reported in Table 2.

¹<https://cran.r-project.org/web/packages/tm/tm.pdf>

Table 2: Corpus statistics

	Corpus 1	Corpus 2
Questions	1	8
Answers	71	2054
Students	90	367
Non-relevant answers ratio	12.7 %	13.8 %

Table 3: Answer filtering results

Method	F_1 scores	
	Corpus 1	Corpus 2
Average Cosine	0.842	0.839
3-words rule	0.250	0.797

4. RESULTS

Figure 2 shows the distribution of average cosines for relevant and non-relevant answers. The non-relevant answers do have an average cosine that is generally close or equal to 0.

The F_1 classification results of the experiments are reported in Table 3. F_1 scores are calculated considering non-relevant answers as Positives, since the filtering aims to recall non-relevant as opposed to relevant answers (note that using relevant answers as Positives would give different scores).

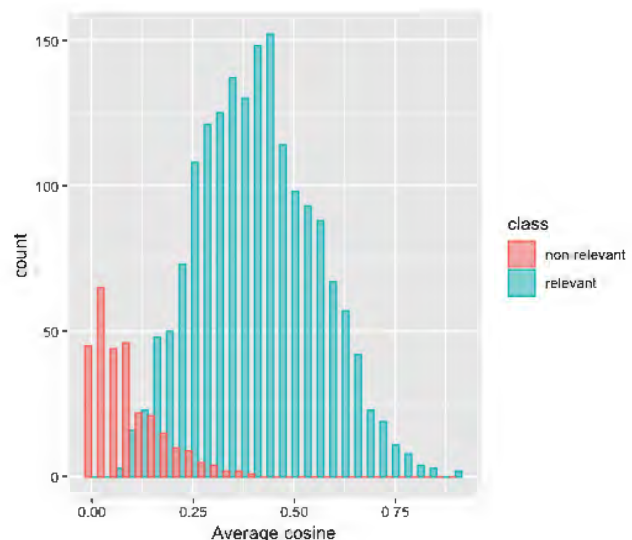
The results are broken down by the methods described and compared with a simple baseline that is actually used in DALITE which consists in classifying answers that contain 3 words or less as irrelevant.

5. CONCLUSION

We propose a simple-to-implement, unsupervised approach to filter out undesired short and open student answers. Undesired answers typically occur in student response systems such as peer learning environments, for eg. DALITE, or in classroom app., for eg. Socrative. The approach is intended to be effective in a context where some answers are collected, but they are not labeled and no correct answer is provided either. Moreover, the approach is intentionally tested with a small dataset because this is also often to be expected in practice. It is compared with a trivial rule based on the number of words that, while parsimonious and effective, misses some undesired answers.

Results show that the approach improves on the simple rule, in particular for Corpus 1 which corresponds to the Socrative context and where the rule performs poorly. A smaller improvement is found for the peer learning environment of Corpus 2. Moreover, the F_1 score shows that not all irrelevant answers are filtered.

Nevertheless, the simplicity of the approach makes it an attractive means to address the undesired answer filtering issue in practice.

**Figure 2: Histogram of average cosines for relevant and non-relevant student answers.**

References

- [1] Sameer Bhatnagar, Nathaniel Lasry, Michel Desmarais, and Elizabeth Charles. Dalite: Asynchronous peer instruction for moocs. In *European Conference on Technology Enhanced Learning*, pages 505–508. Springer, 2016.
- [2] Cindy L Farris and Becky A Salmon. Using classroom response systems to enhance teaching and learning. 2017.
- [3] Lucas Busatta Galhardi and Jacques Duflío Brancher. Machine learning approach for automatic short answer grading: A systematic review. In *Ibero-American Conference on Artificial Intelligence*, pages 380–391. Springer, 2018.
- [4] Robert Kaleta and Tanya Joosten. Student response systems. *Research Bulletin*, 10(1):1–12, 2007.
- [5] J McDonald, RJ Bird, A Zouaq, and Adon Christian Michael Moskal. Short answers to deep questions: supporting teachers in large-class settings. *Journal of Computer Assisted Learning*, 33(4):306–319, 2017.
- [6] Nicole Nawalaniec. Socrative (snowy release). *Journal of the Medical Library Association: JMLA*, 103(4):236, 2015.
- [7] Laszlo Nemeth. Hunspell spell checker and morphological analyzer designed, 2011.
- [8] Stafford R. E. Williams K. M. Reilly, E. D. and S. B. Corliss. Evaluating the validity and applicability of automated essay scoring in two massive open online courses. *The International Review of Research in Open and Distributed Learning*, 15(5):83–98, 2014.
- [9] Dominique L Scapin and JM Christian Bastien. Ergonomic criteria for evaluating the ergonomic quality of interactive systems. *Behaviour & information technology*, 16(4-5):220–231, 1997.

- [10] Md Arafat Sultan, Cristobal Salazar, and Tamara Sumner. Fast and easy short answer grading with high accuracy. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1070–1075, 2016.
- [11] Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010.

Adding duration-based quality labels to learning events for improved description of students' online learning behavior

Matthew Guthrie
University of Central Florida
Lithia Drive
PSB
matthew.guthrie@ucf.edu

Zhongzhou Chen
University of Central Florida
Lithia Drive
PSB
zhongzhou.chen@ucf.edu

ABSTRACT

Many existing studies analyzing log data from online learning platforms model events such as accessing a webpage or problem solving as simple binary states. In this study, we combine quality information inferred from the duration of each event with the conventional binary states, distinguishing abnormally brief events from normal or extra-long events. The new event records, obtained from students' interaction with 10 online learning modules, can be seen as a special form of language, with each "word" describing a student's state of interaction with one learning module, and each "sentence" capturing the interaction with the entire sequence. We used second order Markov chains to learn the patterns of this new "language," with each chain using the interaction states on two given modules to indicate the interaction states on the following two modules. By visualizing the Markov chains that lead to interaction states associated with either disengagement or high levels of engagement, we observed that: 1) disengagement occurs more frequently towards the end of the 10 module sequence; 2) interaction states associated with the highest level of learning effort rarely leads to disengaged states; and 3) states containing brief learning events frequently lead to disengaged states. One advantage of our approach is that it can be applied to log data with relatively small numbers of events, which is common for many online learning systems in college level STEM disciplines. Combining quality information with event logs is a simple attempt at incorporating students' internal condition into learning analytics.

Keywords

Markov chains, online learning modules, log data analysis

1. INTRODUCTION

Understanding and predicting students' learning behavior by mining the log files of online and computerized learning systems has been the focus of a significant body of research in educational data mining. For example, many studies have modeled student learning as a chain of ordered events such as opening a page, viewing a video, or solving a problem [8, 9]. Events are often represented by a binary variable, e.g. whether the student accessed a webpage or answered a problem correctly [8, 14]. While describing events using binary variables can significantly reduce the complexity of the data, doing so can remove important

information from event logs. One indicator of the quality of an event is its duration. For example, abnormally short problem-solving attempts have been associated with either random guessing due to low test-taking effort [4] or answer copying [1]. In two earlier studies [6, 7], we demonstrated that learning events can be separated into "Brief" (B) and "Normal" (N) categories by applying a mixture model clustering algorithm using the time-on-task data alone, since other measurements such as the number of practice problems answered are highly correlated with time-on-task.

We combine duration-based categorical quality labels such as "Brief" or "Normal" with conventional binary event states, such as "Pass" or "Fail," to analyze the log data obtained from students' interaction with 10 Online Learning Modules (OLMs). OLMs are a form of online instructional design in which students progress through learning modules in a pre-determined order [5–7]. Students are required to attempt the assessment problems at least once before accessing the accompanying learning resources in each OLM. The restrictive structure of OLMs has major advantages for data analysis, providing more accurate estimations of duration information. Assessments and learning events are closely coupled on each module; the OLM structure itself improves the interpretability of log data events.

We combine the event logs with categorical quality labels to form a simple artificial language. Each event, such as "Brief Pass" or "Normal Fail," becomes part of a "word" that captures students' interaction with either the assessments or the learning components of a module. Four such "words" form a "phrase" that describes a student's state of interaction with one section of the OLM, and each 10 word "sentence" corresponds to a student's interaction with the entire OLM sequence. Can we gain insight into the patterns in students' learning behavior by understanding the underlying "grammar" of this artificial language? Are there certain combinations of words in these phrases that are frequently followed by other words which indicate that the student is either disengaged or highly engaged with the learning from the OLM sequence? We answer these questions by utilizing a second order Markov chain, a common technique used in natural language processing.

The Markov model can be trained using a relatively small number of events from a student population of approximately 250, owing to the increased information by introducing the quality labels. This advantage is critical for modeling student learning behavior for many STEM disciplines, where solving one problem can take 5 to 10 minutes. A typical weekly online homework assignment of 10 problems may only generate 30 to 50 major events. We demonstrate that it is possible to construct a Markov model using log data from 10 OLMs for a college physics class of approximately 250 students. The resulting Markov chains are visualized to reveal combinations of states that lead to states associated with either disengagement or high levels of engagement in following modules. Engagement is a

Matthew Guthrie and Zhongzhou Chen "Adding duration-based quality labels to learning events for improved description of students' online learning behavior" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 560 - 563

complex concept that has different definitions depending on context and measurement [2]. We adopt a pragmatic definition of engagement to indicate that students spent a normal or extended amount of time (and likely cognitive resources) on consecutive modules, emphasizing the cognitive and behavioral aspect of engagement, bearing similarity to the definitions proposed by Miller [11] and described by Motz, et al. [12].

2. MET ODS

2.1 Structure of OLM and OLM Data

Data analyzed in this study were collected from student interaction with 10 OLMs assigned as homework to be completed over a period of two weeks in a calculus-based college physics class. Students were not required to finish the entire sequence in one session, but the modules must be completed in the order given. As described in detail in several earlier papers [5–7], each OLM consists of an assessment component (AC) and an instructional component (IC). Students are required to attempt the AC at least once before being able to learn from the IC and can make additional attempts after interacting with the IC. Most interactions with each OLM can be divided into three stages: **Pre-Learning**: attempting the AC once or twice before accessing the IC, **Learning**: interacting with the IC after one or two initial failed AC attempts, and **Post-Learning**: making additional attempts on the AC after learning from the IC. If a student passes the AC during the Pre-Learning stage, they will not have the following two stages as the student will immediately proceed to the next OLM.

2.2 Combining Quality Labels with Events

For each Pre-Learning and Post-Learning stage, students' interaction with the AC is captured by the attempt outcomes: "Passing" (P) or "Failing" (F). The quality of each attempt is estimated from the duration of the attempt, t , which is classified into three categories: "Brief" (B: $t < 40$ s), "Normal" (N: $40 \text{ s} \leq t < 180$ s), and "Extensive" (E: $t \geq 180$ s). These cutoffs were determined based on a mixture-model clustering method applied to log-transformed attempt duration data [7]. Combining the quality categories with the attempt outcomes results in six different states: BF, BP, NF, NP, EF, and EP. The EF and EP states are only assigned to the Post-Learning stages, since there were significantly fewer attempts with $t \geq 180$ s in the Pre-Learning stages, and it was less clear whether those longer attempts resulted from longer problem-solving time or students leaving the system. Previous work explains these categories and cutoffs in detail [7].

For the Learning stage, students' interaction with the IC was modeled as a single learning event described by a binary variable. The duration of the learning event is classified as "Brief" or "Normal" according to cutoffs determined for each module by a mixture-model clustering analysis of learning time distribution [7]. An isolated "Brief" category does not necessarily imply that the event is of lower quality. Brief learning can result from a student having a high level of incoming knowledge and only needed to quickly view the learning resources to answer the problem. In a small number of cases, students made 3 or more failed attempts on the AC before accessing the IC or kept attempting the AC until all attempts were used up without accessing the IC. Those cases are classified as "Other." In even fewer cases, due to a corrupted log file or other system glitch, some students were able to proceed to the next module without finishing the current module. Those cases were classified as "NAOther," making a total of 28 possible states, listed in Table 1.

Table 1 All possible interaction states. D Disengaged. E Highly Engaged. See section 2.2 for other acronym definitions.

State	Rank	Indication	State	Rank	Indication
NAOther	0	D	BF-N-EP	14	E
Other	1	D	NP- -	15	E
BP- -	2	D	NF-B-BF	16	
BF-B-BF	3	D	NF-B-BP	17	
BF-B-BP	4	D	NF-B-NF	18	
BF-B-NF	5		NF-B-NP	19	
BF-B-NP	6		NF-B-EF	20	
BF-B-EF	7		NF-B-EP	21	
BF-B-EP	8		NF-N-BF	22	
BF-N-BF	9		NF-N-BP	23	
BF-N-BP	10		NF-N-NF	24	E
BF-N-NF	11		NF-N-NP	25	E
BF-N-NP	12		NF-N-EF	26	E
BF-N-EF	13	E	NF-N-EP	27	E

2.3 Defining States Associated with Either Disengagement or High Engagement

It is difficult to estimate the level of engagement associated with a state, and the same state can be observed from students with different levels of engagement, though there are several states that are more likely to be associated with either a very low or a very high level of engagement with the learning process. For example, "Brief" learning events are more likely than "Normal" learning events to come from students who skimmed through the content [7]. Students displaying consecutive "Brief" events on the same module, such as in state BF-B-BF, are more likely to be disengaged with the learning process.

Consecutive "Normal" or "Extensive" events are more likely to come from students who are highly engaged with learning. While individual "Extensive" events may be caused by a student leaving the computer without exiting from the module, it is much less likely that three such events occur on the same module. We assumed that states with three consecutive "B" labels or BP- - are more likely associated with disengagement. BP- - is included because "Brief" problem solving occurs in under 40 s, which likely resulting from a guessing attempt, or answer copying event [7]. States with three consecutive "N" or "E" labels or NP- - are likely associated with higher engagement. We did not distinguish between productive and unproductive engagement; failed attempts are also included in high engagement states. There are two exceptions to these rules: First, the "Other" state is classified as "Disengaged," since most engaged students should at least look at the instructional resources after two failed attempts. Second, the BF-N-EF and BF-N-EP states are classified as highly engaged, since it is possible that the student quickly decided that the assessment problem was too difficult and immediately engaged in the learning process.

2.4 Training of the Markov Model

We define four sequential states to be a "phrase," a balance between model accuracy and available computational power. Phrases are analyzed in the context of the modules in which they occurred. Ten states are defined as a "sentence" and each student contributed a ten-state sentence to the text corpus. The Markovify Python library [10] was used to parse the text corpus. We utilized second order Markov chains because student interaction on a single module is unlikely to adequately indicate the complexity of their subsequent behavior. The Markov model was used to build second order Markov chains for every combination of initial states in modules 1 and 2, modules 6 and 7, and modules 7 and 8. The Markov chains

were used to investigate student behavior as they progressed through the modules and to infer how changes in behavior can be related to student engagement levels.

3. RESULTS

3.1 Outcomes of Second Order Markov Chain

For a given pair of states on the two input modules, we use the Markov model to return the probability of observing states in the two following modules. We consider a chain as “probable” if the probability of the last two states adds up to more than 100%. On average, 0.2% of all chains generated are considered probable. The three cases for which we ran the Markov model are listed in Table 2, along with the number of probable chains in each case. Those three cases are of particular interest because a previous analysis of the data [7] revealed that more students have lower levels of engagement in modules 3, 8, 9, and 10.

Table 2 Combinations of input states for modules (M1-M10) which were analyzed in this study.

Case	Input	Predict	All chains	Probable chains	Disengaged chains	ighly Engaged chains
I	M1 M2 M3 M4		55664	169	3	66
II	M6 M7 M8 M9		45472	65	18	15
III	M7 M8 M9 M10		72912	108	31	16

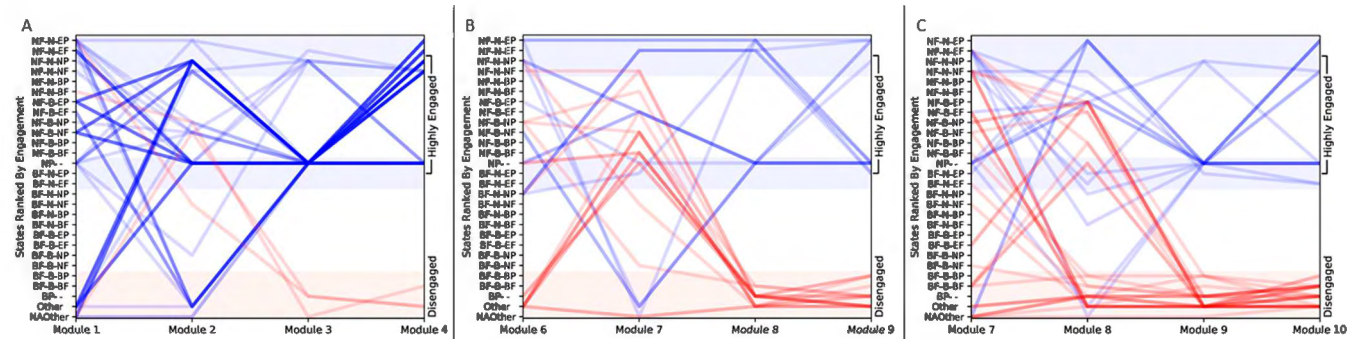


Figure 1 Probable chains that lead to consecutive disengaged states (red) or highly engaged states (blue). Darker lines indicate where multiple chains overlap blue and red zones highlight states associated with disengagement or high engagement, respectively.

Figure 1A (M1-M4) shows significantly more chains leading to highly engaged states on M3-M4 over disengaged states. On M1, those chains started from either a variety of states above NP- -, or from Other and NAOther. On M2, most of the chains concentrated on three states: 25: NF-N-NP, 15: NP- -, and 1: Other. While 25 and 15 were highly populated states in the original text corpus, state 1 was scarcely populated for M2; very few students were consistently disengaged on both M3 and M4, which appear early in the learning sequence and cover less difficult concepts. Figure 1B (M6-M9) shows there were almost an equal number of chains leading to either engagement or disengagement on M7 and M8. Most of the chains leading to disengagement passed through one of the states between 15 and 24 on M7. More than half of those chains started in states 2 and 3 on M6. Several chains leading to high engagement started with high effort states on M6 and passed through Other or NAOther on M7. Every chain (except one) starting from or passing through one of the top three states (25-27) led to high engagement, and every chain starting with disengaged states led to disengagement on M8 and M9. Figure 1C (M7-M10) shows there were more chains leading to disengagement over

3.2 Markov Chains Leading to Consecutive Disengaged or ighly Engaged States

We are interested in chains that could indicate engagement or disengagement with the learning process. We consider a student disengaged from the learning process if their interaction states are indicative of disengagement on the last two modules in a chain. If a student’s interaction states are associated with high engagement on the last two modules, the student is considered as highly engaged. The number of chains that lead to consecutive disengaged or highly engaged states on the last two modules are listed in Table 2. Other chains are not included in this analysis because the relation between engagement and the states on the last two modules were not as clear as the chains included. We plot all the chains indicating disengagement or high engagement for each of the three cases in Figure 1. In each case, the 28 interaction states are arranged according to the order listed in Table 1.

This ordering groups states according to similarities in Pre-learning, Learning, and Post-Learning stages, listed from low to high in the order of “B,” “N,” and “E” in quality labels. States with passing events are assumed to be of higher engagement than those with failing events. States associated with disengagement are placed at the bottom and states associated with high engagement are placed at the top, except for states 13-15 in Table 1. For states near the middle of the pack, the ranking does not reflect the learning effort required for each state, as it is difficult to estimate the level of effort required by 11: BF-N-NP compared to 19: NF-B-NP.

engagement on M9 and M10. The chains leading to disengagement started at a variety of states on M7, and forms two groups on M8. The first group passes through disengagement between states 0 and 4, and the second group passes through states between 15-21. Most of the high engagement chains either started from state 26 on M7 or passed through state 27 on M8.

4. DISCUSSION

By comparing Figure 1A-C we identify four common patterns. **1. Disengagement happens late** chains leading to disengagement occur much more frequently on later modules in the sequence. This type of behavior is expected since the difficulty of the modules increase towards the end, yet each module is worth the same amount of course credit. Therefore, students had less incentive to devote effort on the harder modules. **2. Disengagement-free states** states 25-28 are seemingly “immune” to chains leading to disengagement. In all three cases, only two of those chains pass through these states, while most of the chains leading to high engagement involve those states on at least one input module. Students who spend an extensive amount of effort on one module

are more likely to be more persistent, especially on difficult modules towards the end. **3. Disengagement hot zone** states 15-24 (the white area between two blue bands in Figure 1) seem to be a “hot zone” for chains leading to disengagement in all three cases, especially when the state appeared on the second module in the sequence. Those states have a “Brief” label on either the learning stage or the post-learning stage. A brief event in the learning and post learning stages could indicate subsequent disengagement behavior more effectively than a brief event in the pre-learning stage. **4. -shaped high engagement chains** several chains leading to high engagement started with states beyond 15 and passed through either Other or NAOther on the second module, forming a V-shape on Figure 1C. Even highly engaged students may occasionally display disengaged states on certain modules. It also may suggest that the two “other” states are not always associated with disengagement as previously thought. These patterns can be valuable for development of an intelligent and personalized learning system that recommends different learning resources to appropriate student populations [13]. The patterns can also help instructors prioritize efforts in improving the OLMs.

There are several caveats to be investigated and addressed in future studies. Only three pairs of modules were analyzed. Whether the patterns observed are general to all modules or specific to the selected cases can be answered by analyzing every pair of input modules. We adopted a narrow definition of engagement; students spent an expected or extended amount of time completing each component in a single module. This definition is appropriate for the purposes of the current study, but the relation between time-on-task and engagement should be investigated. The current Markov model “predicts” student behavior based on their interaction states in preceding modules, but students’ decisions to engage or disengage from learning involves complex metacognitive processes influenced by multiple external factors (knowledge, instructional condition, metacognitive skills, and emotional states) [3]. We can achieve more accurate outcomes by including more factors that influence students’ metacognitive processes. This study utilizes simple labels obtained from clustering algorithms on time-on-task data. Future studies should investigate the validity of those labels, and to find new and better-quality indicators for the existing events and new events in other learning systems.

We relied on the restrictive structure of OLMs, providing a regular and simple data structure and allowing for straightforward interpretation of interaction states. Events and quality labels used to generate the artificial language can be obtained from essentially any online learning platform, and more sophisticated Markov models are capable of learning languages with many more irregularities. An extension of this work will be to investigate how the current method can be modified and applied to more common learning systems that are more accessible to the average instructor.

5. ACKNOWLEDGMENTS

We thank the Learning Science and Technology team at the University of Central Florida led by Dr. Francisca Yonekura for developing the Obojobo platform and providing technical support.

6. REFERENCES

- [1] Alexandron, G. et al. 2017. Copying Scale: Using Harvesting Accounts for Collecting Correct Answers in a MOOC. *Computers & Education*. 108, (2017). DOI=<https://doi.org/10.1016/j.compedu.2017.01.015>.
- [2] Azevedo, R. 2015. Defining and Measuring Engagement and Learning in Science: Conceptual, Theoretical, Methodological, and Analytical Issues. *Educational Psychologist*. 50, 1 (Jan. 2015), 84–94. DOI=<https://doi.org/10.1080/00461520.2015.1004069>.
- [3] Azevedo, R. et al. 2018. Understanding and reasoning about real-time cognitive, affective, and metacognitive processes to foster self-regulation with advanced learning technologies. *Handbook of self-regulation of learning and performance (2nd ed.)*. (2018), 254–270. DOI=<https://doi.org/10.4324/9781315697048>.
- [4] Barry, C.L. et al. 2010. Do Examinees Have Similar Test-Taking Effort? A High-Stakes Question for Low-Stakes Testing. *International Journal of Testing*. 10, 4 (2010), 342–363. DOI=<https://doi.org/10.1080/15305058.2010.508569>.
- [5] Chen, Z. et al. 2019. Measuring the effectiveness of online problem-solving tutorials by multi-level knowledge transfer. *2018 Physics Education Research Conference Proceedings* (Jan. 2019), 1–4.
- [6] Chen, Z. et al. 2018. Re-designing the Structure of Online Courses to Empower Educational Data Mining. *Proceedings of 11th International Educational Data Mining Conference* (Buffalo, NY, 2018), 390–396.
- [7] Chen, Z. and Guthrie, M. 2019. Measuring the Effectiveness of Learning Resources Via Student Interaction with Online Learning Modules. *arXiv: 1903.08003*. (Mar. 2019).
- [8] Faucon, L. and Kidzi, . 2016. Semi-Markov model for simulating MOOC students. *Proceedings of the 9th International Conference on Educational Data Mining (EDM)*. (2016).
- [9] Hansen, C. et al. 2017. Sequence Modeling for Analysing Student Interaction with Educational Systems. *Proceedings of the 10th International Conference on Educational Data Mining, Edm 2017, Wuhan, Hubei, China, June 25-28, 2017* (2017).
- [10] Markovify: 2015. <https://github.com/jsvine/markovify>.
- [11] Miller, B.W. 2015. Using Reading Times and Eye-Movements to Measure Cognitive Engagement Using Reading Times and Eye-Movements to Measure Cognitive Engagement. 1520, (2015). DOI=<https://doi.org/10.1080/00461520.2015.1004068>.
- [12] Motz, B. et al. 2019. The validity and utility of activity logs as a measure of student engagement. *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (2019), 300–309.
- [13] Ricci, F. et al. 2015. *Recommender systems: Introduction and Challenges*.
- [14] Sahebi, S. and Brusilovsky, P. 2018. Student Performance Prediction by Discovering Inter-Activity Relations. *Proceedings of the 11th International Conference on Educational Data Mining (EDM)* (2018).

Automatic identification of questions in MOOC forums and association with self-regulated learning

Fatima Harrak
Sorbonne Université
CNRS, Laboratoire
d'Informatique de Paris 6,
LIP6, F-75005 Paris, France
fatima.harrak@lip6.fr

François Bouchet
Sorbonne Université
CNRS, Laboratoire
d'Informatique de Paris 6,
LIP6, F-75005 Paris, France
francois.bouchet@lip6.fr

Vanda Luengo
Sorbonne Université
CNRS, Laboratoire
d'Informatique de Paris 6,
LIP6, F-75005 Paris, France
vanda.luengo@lip6.fr

Rémi Bachelet
Centrale Lille
PRES Université Lille Nord de
France
F-59650 Villeneuve d'Ascq
remi.bachelet@centralelille.fr

ABSTRACT

Discussion forums can be a rich source to analyze students' questions but it can be challenging to find relevant categories of questions. We considered here students' posts from the discussion forum of four editions of a same French MOOC on Project Management. We extended a coding scheme to annotate questions based on their content (course vs. non course) and trained 3 stages of an automatic annotation model. Then we studied the correlation between the nature of the questions asked and students' performance and self-regulation. The results are promising and reveal, for the minority of students active on forums, the possibility to use this feature to better estimate their performance and some of their self-regulation skills based on questions they ask.

Keywords

Student's question, discussion forum, coding scheme, self-regulation, student's performance, MOOC

1. INTRODUCTION

Students' questions play an important role in the learning process and are meaningful for both learning and teaching science [4]. The need for students to ask questions, or to point out errors in the course, are as salient in distance e-learning as they are in a classroom setting, thus emphasizing the importance of discussion forums in online learning and in MOOCs in particular [1]. Forums are not only a place for socialization, but also a place where learning happens, as learners post questions, opinions, and concerns, which are viewed, rated and answered by fellow learners and/or teaching staff. Therefore, we conducted analyses to explore the

nature of questions asked by students in a MOOC forum and particularly tried to see the relationship between those questions and students' performance and self-regulation skills. More particularly, we wanted to answer to the following research questions:

(RQ1) Is it possible to reliably annotate questions extracted from MOOC forum posts according to a fine-grained multi-level coding scheme?

(RQ2) Is there a relationship between the nature of the questions asked on a MOOC and the students' performance and mastery of self-regulated skills?

2. STATE OF THE ART

Studying discussion in MOOC forums is still an ongoing topic of research. Zeng et al. [11] identified the confusion messages by using discussion forum posts derived from large open online courses. Sentiment analysis of MOOC forums discussions can also help in identifying the dropout behavior from students' posts [10].

Researchers have studied students' questions in a variety of educational settings, such as classroom [3], tutoring [7] and online learning environments [9]. Graesser and Person [7] developed a taxonomy of questions asked during tutoring sessions to be used for automatic question generation. Although their taxonomy could be relevant to our work, some categories included high quality 'deep-reasoning questions' and are associated to patterns of reasoning which are difficult to identify automatically.

We also investigated how self-regulated learning (SRL) was used to analyze students' interactions in online environments. Dettori et Persico [6] used a taxonomy of indicators of SRL to analyze directly what kind of students are self-regulated from their messages. Bouchet et al. [2] characterized students via clustering according to their interactions with an intelligent tutoring system fostering SRL.

Fatima Harrak, François Bouchet, Vanda Luengo and Rémi Bachelet "Automatic identification of questions in MOOC forums and association with self-regulated learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 564 - 567

Table 1: Descriptive statistics of the 4 MOOC sessions considered (registration, messages and success)

Session	GDP5	GDP6	GDP7	GDP8
Students reg.	17579	23315	19392	24603
Answered to quiz 1	4842	7537	5951	7998
Bas. certif. obtained	2282	3900	2393	4526
Adv. certif. obtained	503	697	559	589
Nb of posts	7655	10597	12224	14072
Nb of unique posters	2087	4717	3504	4760

3. CONTEXT AND DATASET

We consider in this paper four datasets made of forum messages posted in four different sessions (5 to 8) of the same biyearly French MOOC on project management called GDP (French acronym for project management) held in 2015 and 2016. The MOOC allows participants to obtain a basic certificate (15-25 hours estimated workload), and an advanced one (35-45 hours). Therefore for each participant we can determine two final grades and whether one, both or none of the certificates were obtained. The forum is organized around threads created by the pedagogical team to answer to technical or administrative issues, about homework or course content, *etc.* Table 1 provides some basic statistics on the forum usage and number of students registered.

Additionally, participants to the MOOC are invited to fill an optional (only 0.25% bonus points for filling it) research questionnaire after 2 weeks in the MOOC which included 21 psychometrically evaluated questions evaluating their SRL skills in an online setting [5] using 7-point Likert scales along 4 dimensions: (1) cognitive and metacognitive strategies, (2) procrastination, (3) context adaptation, and (4) peer support. We filtered out the participants who failed to answer to embedded attention check questions (*e.g.* "answer 5 here") to increase the reliability of the data considered.

4. QUESTION CODING SCHEME

To identify the nature of students questions in the forums, we considered a sample of 500 messages from the 4 sessions, randomly divided into 3 sub-samples (200/100/200). We applied 4 categorization steps to define a coding scheme as proposed in another context by Harrak et al. [8].

The raw corpus contains unstructured and noisy messages from students (*e.g.* a message can contain several questions, opinions, answers to issues not course related, *etc.*). We first filtered out the messages from the instructors, those that are a reply to other ones (*i.e.* not the root messages) and the explicitly non-course related topics (*e.g.* thread dedicated to technical issues). The messages were then segmented into several questions (using Python library NLTK) and annotated according to their content. The course-based questions were annotated with the coding scheme from [8] (summarized in upper Table 2) which consists in 4 independent dimensions: a mandatory main one (dimension 1), and 3 optional ones (dimensions 2 to 4). For instance, a question could be a request to re-explain the way something work by providing another example (tagged as Ree on dimension 1, Exa on dimension 2, Man on dimension 3, and nothing on dimension 4, *i.e.* vector [Ree,Exa,Man,0]). The non-course related questions were then annotated according to newly

Table 2: Coding scheme for course-based students' questions (Dim 1 to 4, adjusted from [8]) and for non-course related questions (Dim0)

Code	Question category	Description
Dim1-4: Course-based questions		
Dim1: question type		
Ree	Re-explain / redefine	Ask for an explanation already done in the course material
Dee	Deepen a concept	Broaden a knowledge, clarify an ambiguity or request for a better understanding
Ver	Validation / verification	Verify or validate a formulated hypothesis
Dim2: explanation modality / question subject		
Exa	Example	Example application (course/exercise)
Sch	Schema	Schema application or an explanation about it
Cor	Correction	Correction of an exercise in course/exam
Dim3: explanation type		
Def	Define	Define a concept or term
Man	Manner (how?)	The manner how to proceed
Rea	Reason (why?)	Ask for the reason
Rol	Roles (utility?)	What's the use/function
Lin	Link between concepts	Verify a link between two concepts, define it
Dim4: verification type		
Mis	Mistake / contradiction	Found potential error in course/teacher's explanation
Kno	Knowledge in course	Verify knowledge
Exp	Expected knowledge	Verify expected information in exam or quiz (assessment)
Dim0: Non-course related questions		
Soc	Socialization	Social questions
Adm	Administrative issues	MOOC administration: registration, certificate, <i>etc.</i>
Exa	Exam/ quiz	Ask for assessment modality: notes, format, <i>etc.</i>
Tec	Technical issues	Detect a technical problem and ask for solution
Res	Ressources not found	Ask for not found ressources
Too	Tools	Ask for tools for a task
Pha	Phatic	Question that has no real value or information

defined dimension 0 (*cf.* lower Table 2). Two human annotators made separate independent annotations on each dimension, and their agreement was evaluated using Cohen's Kappa. First the kappa was calculated for the agreement on whether a segment was a question or not ($\kappa = 0.85$) and then on explicit questions only ($\kappa = 0.96$). Then agreement was calculated for the topic of the question (course *vs.* non-course, $\kappa = 0.85$). Finally, kappas were calculated for dim1 to 4 ($\kappa_1 = 0.70$, $\kappa_2 = 0.61$, $\kappa_3 = 0.69$, $\kappa_4 = 0.57$) for course questions and for dim0 for non-course questions ($\kappa = 0.58$).

5. AUTOMATIC ANNOTATION

To annotate the set of questions asked by the students, a semi-automatic tool based on rules and keywords manually weighted was used in prior work to annotate automatically the questions. Although effective (average kappa of 0.70), many questions were not annotated by this tool [8], which led us to develop another automatic tool using machine learning techniques trained on the corpus of questions.

Table 3: Kappa values between automatic annotation and the reference manual annotation

Classifier	SVM	DT	GLM	GBT	K-NN	NB	RI
(1)	0.60	0.91	0.91	0.91	0.80	0.54	0.91
(2)	0.66	0.40	0.62	0.51	0.33	0.47	0.43
(3)	-	0.28	0.35	0.37	0.22	0.21	0.11
(4)	-	0.61	0.63	0.68	0.27	0.07	0
(5)	-	0.30	0.09	0.39	0.05	0.03	0.37
(6)	-	0.50	0.56	0.54	0.14	0	0.26
(7)	-	0.48	0.45	0.47	0.19	0.07	0.35

(-): Not suitable for non binary data

We performed the classical preprocessing steps on the training sample of 1307 segments (500 messages) manually annotated: tokenization, stemming, punctuation removal (except for '?') and stopwords (non-meaningful words) removal. We then extracted all the unigrams and bigrams and counted their occurrences in that sample. Each of the 1307 segments was represented by a binary word vector ('1' if the word is in the segment, '0' otherwise). We finally reduced the number of keywords extracted (high number of keywords compared to the number of segments) to keep the most important and significant ones using a feature selection technique (removing less frequent and correlated unigrams/bigrams).

We designed a 3-stage annotator to identify segments with questions, course *vs.* non-course related questions and the nature of those questions. Overall, 7 classifiers were trained to annotate a segment respectively: (1) into question/non-question; (2) into course/non-course related questions; (3) for non-course related questions, according to dim 0; (4-7) for course-based questions, according to dim 1 to 4. For each classifier we trained models using different machine learning techniques with a 10-fold cross-validation: Support Vector Machine (SVM), Generalized Linear Model (GLM), Gradient Boosted Trees (GBT), Decision Tree (DT), K-NN, Naive Bayes (NB) and Rule Induction (RI), each with various values of hyperparameters. For each classifier, the input was the words vectors representing the segments in terms of keywords, and the label to predict was the value associated to the segment in that dimension. Classifiers (1) and (2) took a binary values and each of the other classifiers took nominal values (varying from 3 to 7, according to the dimension).

We then calculated the Kappa values between the predictions from the classification models and the ground truth values from the manual annotation (*cf.* Table 3). The entire corpus of segments of messages was annotated by the techniques with the highest performance for each classification.

6. RELATIONSHIP BETWEEN QUESTIONS, SUCCESS AND SELF-REGULATION

6.1 Data coding

To study the relationship between question type and success and self-regulation, we coded for GDP8 students who posted on the forum the number of segments categorized as :

- an explicit question (NbQ)
- not a question or an implicit one (NbNQ)
- a non-course question (NbQ-NC)
- a non-course question corresponding to a socialization (NbQ-NC-Soc), an administrative issue (NbQNC-Adm), an exam

- (NbQ-NC-Exa), a technical issue (NbQ-NC-Tec), a resource not found (NbQ-NC-Res), a tool issue (NbQ-NC-Too) or a phatic (NbQ-NC-Pha),
- a course question (NbQ-C),
- a course question about a reexplanation (NbQ-C1-Ree), a request to go deeper in a concept (NbQ-C1-Dee), or a verification (NbQ-C1-Ver)
- a course question requesting an example (NbQ-C2-Exa), a schema (NbQ-C2-Sch), or a correction (NbQ-C2-Cor),
- a course question asking for a definition (NbQ-C3-Def), the way to proceed (NbQ-C3-Man), the reason for something (NbQ-C3-Rea), the role of something (NbQ-C3-Rol), or the link between two concepts (NbQ-C3-Lin),
- a course question asking for a verification regarding an apparent mistake (NbQ-C4-Mis), knowledge from the course (NbQ-C4-Kno) or whether something is expected to be learned or not for the assessment (NbQ-C4-Exp).

In addition to those variables relative to the questions asked, we also have for each student :

- **four SRL scores**, measured by a questionnaire [5]. Although the authors average the 4 individual scores into an overall SRL score (with procrastination coded in a reverse manner), we believed they captured different facets of SRL which could individually be associated to different question asking behavior. Therefore we defined for each student their lack of procrastination score (Sc_{ONPr}), their context score (Sc_{Ctxt}), their strategy score (Sc_{OStr}) and their peer support score (Sc_{OPee}). Each score is an average of 5 to 6 questions, between 1 and 7,
- **two performance scores** for the basic/advanced MOOC track (Sc_{Bas} and Sc_{Adv}), a value between 0 and 100.

6.2 Correlation analysis

Method: We calculated for each question variable (NbQ-*) its Pearson correlation coefficient (r) with each of the self-regulation and performance variables (Sc_{*}).

Results: 286 students posted at least one message with a segment containing an explicit question. The results (not all detailed here) reveal that asking explicit questions (on the basic [$p = .012, r = .148$] and the advanced tracks [$p = .000, r = .237$]), and questions on a topic relevant to the course (on the basic [$p = .010, r = .153$] and the advanced tracks [$p = .000, r = .253$]), is a behavior positively correlated with the performance. The questions the most strongly positively correlated to performance are the ones to check one's understanding (Ver) regarding a theme of the course ($p = .000, r = .292$) or a skill one is expected to master for the final exam ($p = .000, r = .282$).

123 students among those who posted at least one message with an explicit question had also filled the SRL questionnaire. The results (summarized in Table 4) reveal that procrastination is not correlated with any particular type of question. It is however logically negatively correlated with the score in the basic ($r = -.349$) and advanced ($r = -.372$) tracks of the MOOC, *i.e.* students who procrastinate have lower scores overall. The context facet of SRL is positively correlated only with the number of messages not containing a question (NQ). The two other facets are more interesting, as the students who self-report being good at using cognitive and metacognitive strategies (such as note-taking) while

Cat.	$ScONPr$		$ScOCtx$		$ScOSTr$		$ScOPee$	
	r	p	r	p	r	p	r	p
NQ	-.064	.480	.178	.049*	-.137	.131	.259	.004*
Q	-.064	.485	.056	.541	.169	.061	.247	.006*
NC	-.104	.254	.123	.174	-.161	.076	.212	.019*
NC-Soc	-.095	.295	.081	.376	-.107	.239	.198	.028*
NC-Adm	-.058	.520	.162	.074	.018	.840	.258	.004*
NC-Exa	-.090	.322	-.006	.951	-.216	.016*	.078	.392
NC-Tec	-.060	.513	.023	.802	.001	.995	.155	.087
NC-Res	.067	.459	.138	.128	.039	.665	.166	.067
NC-Too	.007	.939	.103	.257	.016	.858	-.050	.584
NC-Pha	-.018	.844	.079	.385	-.226	.012*	.093	.307
C	-.028	.761	.005	.958	-.143	.115	.222	.014*
C1-Ree	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
C1-Dee	-.040	.660	.075	.407	-.001	.987	.219	.015*
C1-Ver	-.020	.660	-.022	.805	-.179	.048*	.196	.030*
C2-Exa	.005	.957	.123	.174	-.123	.174	.068	.487
C2-Sch	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
C2-Cor	-.105	.247	.038	.675	-.045	.619	.118	.193
C3-Def	.005	.957	-.130	.151	.135	.136	.055	.547
C3-Man	-.089	.329	.111	.221	-.002	.984	.187	.038*
C3-Rea	-.068	.457	.140	.123	.024	.796	.222	.014*
C3-Rol	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
C3-Lin	-.009	.918	.117	.197	-.067	.459	.076	.406
C4-Mis	-.112	.216	-.063	.491	-.067	.460	-.048	.599
C4-Kno	-.009	.919	-.048	.597	-.176	.052	.191	.034*
C4-Exp	-.024	.795	.097	.288	-.136	.135	.169	.062

n/a: no segment annotated with this code

Table 4: Correlation between the question types and the four SRL scores

learning online are asking less question about the organization of the final exam, less phatic questions and less verification questions. Thus it seems that being more organized, maybe when they watch the video or go through pages of contents, they have a lesser need to verify information probably already mentioned somewhere. As for students who self-report being good at interacting with others to learn in a more efficient manner, logically they post more messages (both questions and non-question), which can be related to the course or not. When analyzing the nature of the questions they ask, they socialize more with others and ask more administrative questions. They also tend to ask very practical questions about the course about how to perform a task or the reason some concept is working that way.

7. DISCUSSION AND CONCLUSION

We have shown it is possible to annotate not only messages from a MOOC forums, but individual questions within sometimes long messages. Segmenting messages allows to distinguish finer-grain intent of the student, using an adapted coding scheme for both course and non-course related questions. This result opens the way to automatically tagging MOOC posts, for instance to help the pedagogical team to quickly know the intent of the messages that have not received a reply yet. Another interesting aspect is the fact that the nature of the questions asked within the messages provides information on some aspects of students' level of self-regulation (their tendency to interact with others for learning and their use of cognitive and metacognitive strategies). It is also worth noting that some of the patterns found here, such as the fact that students who ask verification questions tend to succeed overall better than others, are consistent with previous results in a different context [8].

Some limits include: the topic of the MOOC which hindered the classifiers performance with its low technical vocabulary (words overlap between the content and context of the course) and the average kappa values obtained for the classifiers which can reduce the impact of some correlations observed, correlation values which are themselves never extremely high even when $p < .05$. Finally, as always with results relative to MOOCs forum, they are only used by a minority of active learners. Future directions involve considering some messages excluded here (messages that are not root in the thread, technical or socialization threads which could fit in the non-course coding scheme), and considering forums from MOOCs on different themes to build up a larger corpus of messages, to try to improve the annotator performance.

8. REFERENCES

- [1] M. A. Andresen. Asynchronous discussion forums: success factors, outcomes, assessments, and limitations. *J. of Educ. Technology & Society*, 12(1):249–257, 2009.
- [2] F. Bouchet, J. M. Harley, G. J. Trevors, and R. Azevedo. Clustering and Profiling Students According to their Interactions with an Intelligent Tutoring System Fostering Self-Regulated Learning. *J. of Educ. Data Mining*, 5(1):104–146, 2013.
- [3] C. Chin and G. Kayalvizhi. Posing Problems for Open Investigations: What questions do pupils ask? *Research in Science & Technological Education*, 20(2):269–287, 2002.
- [4] C. Chin and J. Osborne. Students' questions: a potential resource for teaching and learning science. *Studies in science education*, 44(1):1–39, 2008.
- [5] L. Cosnefroy, F. Fenouillet, and J. Heutte. Développement et validation d'une échelle d'apprentissage autorégulé en ligne. In *2e Colloque international e-Formation des Adultes et Jeunes Adultes*, Lille, France, 2018.
- [6] G. Dettori and D. Persico. Detecting Self-Regulated Learning in Online Communities by Means of Interaction Analysis. *IEEE Transactions on Learning Technologies*, 1(1):11–19, 2008.
- [7] A. C. Graesser and N. K. Person. Question asking during tutoring. *American educational research journal*, 31(1):104–137, 1994.
- [8] F. Harraq, F. Bouchet, V. Luengo, and P. Gillois. Profiling Students from Their Questions in a Blended Learning Environment. In *Proc. of the 8th Int. Conf. on Learning Analytics and Knowledge, LAK '18*, 102–110, New York, NY, USA, 2018. ACM.
- [9] H. Li, Y. Duan, D. N. Clewley, B. Morgan, A. C. Graesser, D. W. Shaffer, and J. Saucerman. Question Asking During Collaborative Problem Solving in an Online Game Environment. In *Intelligent Tutoring Systems, LNCS*, 617–618. Springer, Cham, 2014.
- [10] M. Wen, D. Yang, and C. P. Rosé. Sentiment Analysis in MOOC Discussion Forums: What does it tell us? In *Educational Data Mining* 130–137, 2014.
- [11] Z. Zeng, S. Chaturvedi, and S. Bhat. Learner Affect Through the Looking Glass: Characterization and Detection of Confusion in Online Courses. In *Int. Conf. on Educational Data Mining*, 272–277, 2017.

Students' Use of Support Functions in DBAs: Analysis of NAEP Grade 8 Mathematics Process Data

Juanita Hicks
American Institutes for Research
1400 Crystal Dr.
Arlington, VA 22202
1 (202) 403-5446
jhicks@air.org

Ruhan Circi
American Institutes for Research
1400 Crystal Dr.
Arlington, VA 22202
1 (202) 403-5503
rcirci@air.org

Mengyi (Elaine) Li
American Institutes for Research
2800 Campus Dr. Suite 200
San Mateo, CA 94403
1 (650) 376-6368
mli@air.org

ABSTRACT

To address the changing landscape in educational assessment, the National Assessment of Educational Progress (NAEP) transitions to digitally-based assessments (DBAs). DBAs permit the collection of data about how students progress through assessments, which was not feasible with paper-based assessments. The exploration of process data is a relatively new field, with most current studies focusing on response time analyses. This study will incorporate other process data information, such as the frequencies and durations of support functions used related to students' cognitive processes. Using data from one released block of the NAEP 2017 Mathematics Assessment for Grade 8, this study aims to provide an understanding of how students' use of support functions (e.g. drawing and highlighting) are related to response time (RT) and mathematics achievement.

Keywords

Process data, Digitally-based assessments, NAEP, Support functions.

1. INTRODUCTION

The study of response processes was explicitly stated as one source of validity evidence in the 2014 Standards for Educational and Psychological Testing [1]. Response processes are the actions or thought processes that test takers demonstrate when engaging with assessments. Think-aloud protocols, interview sessions, eye-tracking technology, and response time procedures are all common ways of capturing such processes. Since the advent of large-scale, digitally-based assessments (DBAs), the recording of item response time (RT) has resulted in an increased number of research studies investigating a wide range of topics, such as the development of psychometric models [7, 8], modeling of response speed and accuracy [3], student motivation [11], and the relationship between response time and item- and person-level factors [6].

Yet, exploration of specific functions and their relationship with

item response time (RT) has been sparse. With the availability of process data, students' use of support functions such as drawing, and highlighting have also been recorded. The collection of this ancillary data in modern assessments can lead to potential advances in the type of research normally reserved for assessment data. To our knowledge, there has not been a study focusing on use of support functions, in the context of a digitally-based mathematics assessment. However, literature in other contexts (e.g. classroom assessments for reading and science subjects) has highlighted the connection between students' support-related cognitive processes and better learning outcomes [2, 4, 9]. Undergirded by the cognitive theory of multimedia learning [7], the effectiveness of drawing (i.e. learner-generated representation of provided material) was confirmed [10]. Similarly, students' navigation behavior (i.e. selection of task-relevant hypertext pages) was found to be the key to understanding the variation in student comprehension of digital texts [4], while students' highlighting behavior was found to play a role in encoding and organizing information [5].

For the current study, response processes are defined as the actions that students demonstrate, using available digital tools when responding to test items on NAEP. NAEP is the largest nationally representative assessment of what America's students know and can do in various subject areas. NAEP is administered to 4th, 8th, and 12th grade students, every two years in various subjects. Specifically, the following research questions will be examined in the present study:

- 1) How often do students display support functions in the NAEP mathematics assessment? How do support functions differ by item type?
- 2) What is the relationship between support functions and RT?
- 3) Can distinct groups of students with unique patterns of support functions be identified?
- 4) Do clusters, (i.e., identified groups), have a relationship with any prominent observed variables?
 - a. What is the prevalence of male and female students in these clusters?
 - b. What is the prevalence of different race/ethnicity groups in these clusters?
 - c. What is the prevalence of different ELL and IEP status in these clusters?

Juanita Hicks, Ruhan Circi and Mengyi Li "Students' Use of Support Functions in DBAs: Analysis of NAEP Grade 8 Mathematics Process Data" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 568 - 571

- d. Is there a difference in mathematics performance between clusters?

2. METHODS

2.1 Data

The current study used a single block of items (i.e., Block MC) that was recently released from the NAEP 2017 Grade 8 Mathematics assessment. The block included 25,300 8th grade students. The sample was selected using a two-stage, stratified random-sampling design, with schools selected in the first stage, and students selected in the second stage. The assessment was administered to students using a balanced-incomplete-block spiraling design, in which test booklets for each student contained two of 15 blocks of cognitive items. The maximum amount of time students could spend on each block was 30 minutes. Students with extended time accommodations and interrupted sessions were excluded during the data cleaning process in the current study.

The focused block comprised six item types and a total of 15 items: a) two multiple-choice (MC) items, b) three composite constructed response (Composite CR) items, c) two multiple-selection (MS) grid items, d) five matching items, e) one zone item, and f) two fill-in-the-blank items. Each item produced student actions captured in the process data. There were nearly 13 million instances of actions produced from 37 unique functions (e.g. drawing, highlighting, and navigating).

2.2 Analysis

To examine the relationship between support functions, item response time (RT), and mathematics achievement, a preliminary list of support functions was created based on data collected in NAEP 2017 Grade 8 Mathematics process data. After a descriptive exploration, 12 out of 37 functions were selected to be included in the study (Table 1). Student response time per item was calculated, and results were aggregated to get the total response time for each student. The relationships were explored for all support functions and time usage via descriptive methods.

To identify heterogeneous use of support functions in preliminary

analyses, students were clustered based on their use of support functions via a model-based clustering approach that identifies heterogeneity in the population based on the observed data (e.g., Everitt, 2005), using the EM algorithm for maximum likelihood estimation. Subpopulations were modeled separately, and the overall population was modeled as a weighted sum of those subpopulations using finite mixture models. NAEP scale scores for each examinee are reported as a set of 20 plausible values (PVs) that represent a distribution of possible scores and apply to students from measured population groups. Instead of individual student scores, plausible values describe the situation where true scale scores describing the underlying performance for each student are unknown. To examine student performance, we used plausible values (PVs) at the theta scale as predicted ability based on the performance across two blocks (instead of per block).

3. Results

3.1 Support Functions in NAEP Mathematics

To address RQ1, we explored how often students displayed support functions in the NAEP grade 8 mathematics assessment. Frequencies of the actions were calculated overall and per item type. The overall frequencies for the 12 support functions are presented in Table 1, together with the percentage of students who used the support function at least once.

Table 1 shows that students displayed varied use of support functions. For example, functions related to screen orientation, such as vertical item scroll and focusing were used at least once by almost all students. The navigation function was used by 75% of students, while functions like draw and highlight were used much less often by 52% and 19% of students, respectively.

However, some of the support functions were expected to be used more frequently for specific types of items. Therefore, the percentage of students that used each function at least once per item type was also examined (see Figure 1).

As expected, Figure 1 shows that some of the functions were only available (or used) for specific item types. For example, Focus (i.e., Receive Focus and Lose Focus) was only activated for composite constructed response and fill-in-the blank items, as this

Table 1. Frequency Use of Support Functions

Support Function	Description	Frequency	Percent of Students Used Function at Least Once
Draw	Student finished drawing with the scratchwork draw tool	900597	52
Vertical Item Scroll	Student scrolls an item which has a vertical scrollbar	557856	99
Receive Focus	When a response entry field in a discrete item receives focus	521631	99
Lose Focus	When a response entry field in a discrete item loses focus	509976	99
Click Progress Navigator	When the Student selects a tab in the Progress Navigator for discrete items	191521	76
TextToSpeech	Recorded when the student turns test to speech/read aloud more on or off	161829	30
Erase	When the Student finishes erasing with the Scratchwork Erase tool	88205	33
Equation Editor Button	When the Student selects a button in the Equation Editor tool	83993	38
Highlight	When the Student finishes highlighting with the Scratchwork Highlight tool	58972	19
Clear Scratchwork	When the Student selects the Clear Scratchwork button of the Scratchwork tool	37723	34
Open Equation Editor	When the Student selects the Equation Editor button to toggle the Equation Editor tool on	20669	40
Horizontal Item Scroll	Recorded when the Student scrolls an item which has a horizontal scrollbar	3414	3

function was associated with text entry.

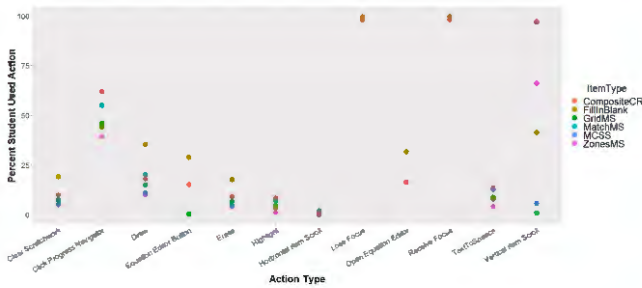


Figure 1. Percentage of Students Using Support Functions by Item Type.

Similarly, equation editor was used for the same types of items but by much fewer students (15% for CompositeCR and 31% for Fill-In-Blank). The difference between Open Equation Editor and Equation Editor Button showed that some of the students opened the editor but did not click on anything in the editor for GridMS items. Figure 1 also shows that students used the navigation function the most on constructed response items. These items were probably more difficult or required more input from students, which is why students used navigation more on these items.

Although Draw and Highlight were used by less students, for those that did use the tools, results showed that they were used across all item types. To understand the use of the Draw tool, this finding will be connected with how much time students spent on this support function. For all other functions, some variation was observed in the percent use of support functions by item types, signaling that students may be using the support function features in different ways.

3.2 Support Functions and Total RT

To address RQ2, we explored the relationship between support functions and total response time (RT). We first calculated the frequency of the use of the support functions for each individual student by item. A summary of the frequency of each support function is presented in Table 2. For total RT, we summed the time spent across all items.

Table 2. Distribution of Support Function Frequency Use

Interpretation	1 st Q.	Mdn	Mean	3 rd Q.	Max	SD
Draw	0	#	35.6	50	1290	63.7
Vertical Item Scroll	10	20	22.1	30	200	15.6
Receive Focus	10	20	20.6	30	190	12.4
Lose Focus	10	20	20.2	20	190	11.8
Click Progress Navigator	#	#	7.6	10	100	8.9
TextToSpeech	0	0	6.4	#	570	21.8
Erase	0	0	3.5	#	490	10.7
Equation Editor Button	0	0	3.3	#	850	13.1
Highlight	0	0	2.3	0	350	9.8
Clear Scratchwork	0	0	1.5	#	140	3.9
Open Equation Editor	0	0	0.8	#	20	1.3

Horizontal Item Scroll	0	0	0.1	0	240	2.5
# Rounds to zero.						

As noted in Table 2, not all students used the support function tools. Therefore, the minimum number of the action frequency is zero for all functions. Table 2 further shows that individual students vary with the use of support functions.

Next, we explored the relationship between the use of support functions and total response time. As an example, we selected the drawing function (which was used by half of the students) and we explored the total response time between two groups of students: a) those that did not use the drawing tool and b) those that used the drawing tool. Figure 2 shows that after 22nd minute, there were more students who used the drawing tool than those who did not. A similar result was found for the highlight function. These results suggest that on the latter portion of the block, there are more students who use these tools, possibly because of a higher concentration of difficult items where support functions would be useful.

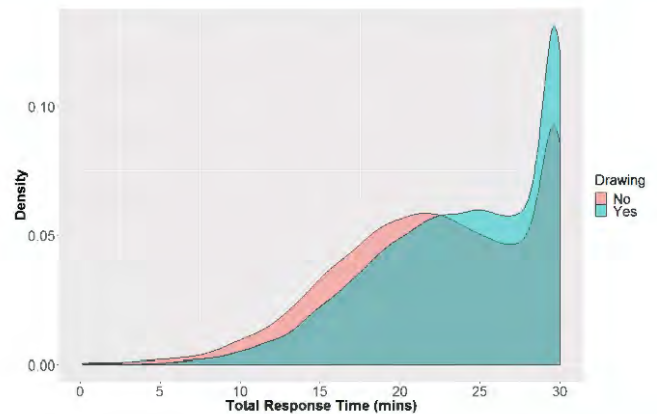


Figure 2. Total Response Time Used by Student Groups: Used or Not Used Drawing Tool.

3.3 Clustering Support Function Behavior

To address RQ3, we also explored if it was plausible to cluster students based on their support function behaviors (i.e., frequency). For this purpose, we standardized all support function frequencies (if students did not use a function at all, they have value of zero). We used only 10 of the support functions for clustering (we excluded Lose Focus and Open Equation Editor as they have almost the same function as Receive Focus and Equation Editor Button, respectively).

Our preliminary exploration led to a 3-profile solution via a model-based clustering algorithm. Figure 3 shows the clustering results for all students. The first extracted profile was represented by hardly any use of support functions. The second extracted profile was a composition of Receive Focus, Equation Editor Button, and Vertical Item Scroll, which provided evidence of students' monitoring processes operationalized by navigating [4] and modifying equations. The third extracted profile was a composition of Highlight, Draw, Erase, and Clear Scratchwork, which provided evidence of students' cognitive processes where they construct mental representations of the test content [7,10]. These results suggest that students' use of support functions can be classified into meaningful profiles, which are potentially related to different aspects of cognitive processes.

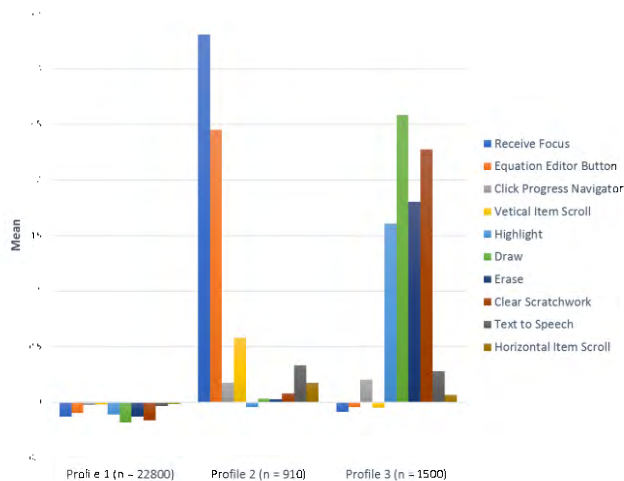


Figure 3. Profiles of Student Support Functions.

To address RQ4, we examined the profiles to see if they had a relationship with any prominent observed variables such as gender, race/ethnicity, ELL, and IEP status. Profiles were also examined to see if there were any performance differences.

For all observed variables and performance, there were no substantial differences between profiles, except in the case of gender, where there were more female students in profile one (no support function use) compared to male students.

Table 3. Distribution of Demographics by Profile

Profile	Profile 1	Profile 2	Profile 3
Gender			
Male (%)	34	56	50
Female (%)	66	44	50
Race/Ethnicity			
>1 Race, not Hispanic (%)	3	‡	‡
African American, not His (%)	16	17	16
Amer Ind/Alsk Nat (%)	2	‡	‡
Asian, not Hispanic (%)	5	‡	5
Hispanic of any race (%)	22	21	20
Native Ha/ Pac Island (%)	1	‡	‡
White, not Hispanic (%)	52	51	53
ELL			
No (%)	94	94	94
Yes (%)	4	‡	‡
Formerly ELL (%)	1	‡	‡
Performance			
Mean PV	-0.002	-0.001	0.050
Median PV	-0.024	-0.016	0.030

‡ Reporting standard are not met (too few cases).

4. CONCLUSIONS

Process data helps us to extract more information about how students interact with available digital tools when responding to

test items in digitally-based assessments. By exploring process student support functions, this study helps a variety of stakeholders, understand student cognitive processes when taking mathematics assessments. This would allow researchers to better understand the actual use of support functions by certain groups of students (e.g., those performing at the below basic level). Interestingly, the majority of students did not use the available support functions, which may be due to unfamiliarity with technology and could lead to a cold start (where students did not have enough information and time to learn the system tools). In all, process data on student support functions is an under-researched area with promise, and our study attempts to fill the research gaps in this field.

5. REFERENCES

- [1] American Educational Research Association, American Psychological Association, and National Council on Measurement in Education (2014). *Standards for Educational and Psychological Testing*. Washington, D.C.: American Educational Research Association.
- [2] Dunlosky, J., Rawson, K. A., Marsh, E. J., Nathan, M. J., and Willingham, D. T. (2013). Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology. *Psychological Science in the Public Interest*, 14(1), 4-58.
- [3] Goldhammer, F. (2015). Measuring ability, speed, or both? Challenges, psychometrics solutions, and what can be gained from experimental control. *Measurement: Interdisciplinary Research and Perspectives*, 13(3-4), 133-164.
- [4] Hahnel, C., Goldhammer, F., Naumann, J., and Kröhne, U. (2016). Effects of linear reading, basic computer skills, evaluating online information, and navigation on reading digital text. *Computers in Human Behavior*, 55, 486-500.
- [5] Li, L., Tseng, S., and Chen, G. (2016). Effect of hypertext highlighting on browsing, reading, and navigational performance. *Computers in Human Behavior*, 54, 318-325.
- [6] Masters, J., Schnipke, D. L., and Connor, C. (2005, April). *Comparing item response times and difficulty for calculation items*. Paper presented at the annual meeting of the American Educational Research Association, Montréal, Canada.
- [7] Mayer, R. E., and Moreno, R. (1998). A cognitive theory of multimedia learning: Implications for design principles. *Journal of Educational Psychology*, 91(2), 358-368.
- [8] Ranger, J. (2013). Modeling responses and response times in personality tests with rating scales. *Psychological Test and Assessment Modeling*, 55(4), 361-382.
- [9] van der Linden, W. J., and van Krimpen-Stoop, E. (2003). Using response time to detect aberrant responses in computerized adaptive testing. *Psychometrika*, 2, 251-265.
- [10] van Meter, P. and Garner, J. (2005). The promise and practice of learner-generated drawing: Literature review and synthesis. *Educational Psychology Review*, 17(4), 285-325.
- [11] Wise, S. L., and Kong, X. (2005). Response time effort: A new measure of examinee motivation in computer-based tests. *Applied Measurement in Education*, 18(2), 163-183.

Investigating Writing Style Development in High School

Stephan Lorenzen
University of Copenhagen
lorenzen@di.ku.dk

Niklas Hjuler
University of Copenhagen
Hjuler@di.ku.dk

Stephen Alstrup
University of Copenhagen
alstrup@di.ku.dk

ABSTRACT

In this paper we investigate writing style development among Danish high school students. More than 10K students with more than 130K essays are analyzed. Writing style itself is often studied in the natural language processing community, but usually with the goal of verifying authorship, assessing quality or popularity, etc.

In this work, we analyze writing style changes over time, with the goal of detecting global development trends among students, and identifying at-risk students. We train a Siamese neural network to compute the similarity between two texts. Using this similarity measure, a student's recent essays are compared to their first essays, and a writing style development profile is constructed. We cluster these student profiles and analyze the resulting clusters. We evaluate clusters with respect to writing style quality indicators, and identify optimal clusters, showing significant improvement in writing style, while also observing suboptimal clusters, exhibiting periods of limited development and even setbacks.

Furthermore, we identify general development trends between students, showing that as students progress through high school, their writing styles deviate, leaving students less similar when they finish high school, than when they start.

Keywords

Student clustering, Writing style analysis, Siamese Neural Network, Educational Systems

1. INTRODUCTION

One of the most essential skills, learned during the course of primary, secondary and high school, is writing. While the main focus of primary school are on basic writing skills, high school will be more focused on improving *the linguistic writing style* of a student, i.e. the quality of the written text as perceived by the reader. While the definition of quality in linguistic writing style is discussed [7], several measures are correlated to writing style being perceived as good, for instance use of vocabulary, sentence structure and readability

[5]. Our main focus is writing style *development* through the course of high school, while writing style quality will have a secondary role. We consider data from Danish high schools, consisting of Danish essays, and investigate the general development patterns among the students during the three years of study. The data is supplied by MaCom¹, the company behind the learning management system Lectio, used by 90% of Danish high schools. We identify patterns among thousands of students across different classes and institutions, allowing us to provide teachers with new insights, which the data available to the teacher might not show.

Our method allows for identifying students with deviating writing style development, or sudden significant changes in writing style, which could indicate cheating. Furthermore, we compute several *text quality* measures for the different development profiles, in order to determine whether a profile indicates improvement. Our approach is based on methods from authorship verification; we use a Siamese neural network for learning a writing style similarity measure. Using this measure, we generate and cluster writing style development profiles. The found clusters are then analyzed, in order to determine optimal and suboptimal development patterns. Our analysis of the MaCom data shows, how writing style changes during high school, supporting conclusions from the literature. [1, 2, 9]. While this paper presents a case study of the data from MaCom, the methods used for analysis are of independent interest, and not specific to the Danish language or high school.

Writing style analysis, has been studied in the natural language community for many years. Typically, the analysis of writing style is used as a middle link for tasks such as *authorship verification* [6, 8, 9], in which the claimed author of a text must be verified based on available previous work by said author. Some studies investigate the quality of writing, for instance prediction of popularity of news articles [10], or the quality of scientific articles [3]. Few studies consider development of writing style as the main objective, e.g. [1] which considers the change of writing style of two famous Turkish writers. Finally, several studies related to writing style have been conducted using the data available from MaCom. [2] investigates temporal aspects of authorship attribution while [9] applies neural network based authorship verification methods to the data.

Niklas Hjuler, Stephan Lorenzen and Stephen Alstrup "Investigating Writing Style Development in High School" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 572 - 575

¹The data set is proprietary and not publicly available

2. METHODS AND SETUP

We describe in this section our method for tracking writing style development among students over time. Below, \mathcal{A} denotes the set of students, $\alpha \in \mathcal{A}$ denote a single student with texts $t \in T_\alpha$, while $T = \cup_{\alpha \in \mathcal{A}} T_\alpha$ denotes the entire corpus of texts. As we wish to compare writing style between different essays, we need a writing style similarity measure $s : T \times T \rightarrow [0, 1]$, which will be further utilized in the analysis. Primarily, we focus on determining development patterns by generating a *writing style development profile* P_α for each student α . These profiles are then clustered and analyzed with respect to different measures for text quality. Finally, we also explore how the similarity between random students change depending on their current progress through high school. This is done by sampling random pairs of texts $t_1 \in T_\alpha, t_2 \in T_\beta$ and computing their similarity. We then consider how the similarity changes depending on if α and β are in the same grade or not.

We utilize a Siamese neural network for computing the similarity $s(t_1, t_2)$ between two texts t_1 and t_2 . We considered several different architectures, using different input channels (e.g. char, word, part of speech tags), testing each option using a validation set. The optimal architecture was found to be the same as used in [9], which was evaluated on the same data set. The architecture uses only character level input. The texts are first encoded in the *encoding part* of the network, consisting of a character embedding (dimension 5), followed by two parallel convolutions (kernel size 4 and 8 with 500 and 700 filters respectively) and global max pooling layers, producing an encoding of size 1200. Encodings for t_1 and t_2 are then passed to the *decider part* of the network, which first computes the absolute difference between the encodings, before applying four 500 neuron dense layers using the ReLU activation function and a dropout rate of 0.3. Finally, a softmax layer with two neurons produces the final output.

The writing style profile P_α for student α is then computed by first determining their initial writing style. This is done by considering the early work of a student, consisting of the first m essays. P_α then consists of a chronologically ordered sequence of similarities, between any $t \in T_\alpha$ and these m essays:

$$p_i = \frac{1}{m} \sum_{j=1}^m s(t_i, t_j),$$

where t_i denotes the i 'th text handed in by the student. Note, that p_1, p_2, \dots, p_m are not independent, and thus we exclude the first $m - 1$ texts, and re-index such that $p_j = p_{i-m+1}$. Furthermore, for each text, we let τ_j denote the time in months since t_m was written, i.e. the time since p_0 , with $\tau_0 = 0$. Now, the final profile becomes the sequence consisting of pairs (τ_j, p_j) of length $|T_\alpha| - m + 1$.

These profiles are now clustered using a slightly modified k -means clustering. Before clustering, for each profile P_α , an approximate profile \hat{P}_α is constructed by interpolating values between any two consecutive pairs (τ_j, p_j) and (τ_{j+1}, p_{j+1}) , in intervals of 0.05 months. Thus \hat{P}_α becomes a vector $\hat{P}_\alpha \in [0, 1]^{\ell_\alpha}$ consisting of similarities for every 0.05 month, with length ℓ_α .

Data set	#students	#texts	#SIM
T_{train}	5418	70432	934720
T_{val}	989	12997	173536
$T_{analyze}$	3688	47666	N/A
Total	10095	131,095	1108256

Table 1: Data set overview.

These approximate profiles are then clustered. The clustering is complicated by profiles having variable length: \hat{P}_α has length ℓ_α depending on $\tau_{|T_\alpha|-m+1}$ (the time span between t_m and t_{T_α}), specific to α . Hence, distance computation used in the clustering algorithm is modified slightly; we compute the distance $dist(\hat{P}_\alpha, \hat{P}_\beta)$ between two profiles \hat{P}_α and \hat{P}_β by computing the Euclidean distance between the prefixes of length $\ell = \min\{\ell_\alpha, \ell_\beta\}$ of the two profiles:

$$dist(\hat{P}_\alpha, \hat{P}_\beta) = dist_E(\hat{P}_\alpha[1..\ell], \hat{P}_\beta[1..\ell]),$$

where $dist_E$ denotes the Euclidean distance, and $v[1..n]$ denotes the prefix of length n of vector v .

Similarly, when computing centroid C_r for cluster \mathcal{C}_r , profile \hat{P}_α contributes only to the ℓ_α first entries of C_r . Thus, with $\mathcal{C}_r^j = \{\hat{P}_\alpha | \hat{P}_\alpha \in \mathcal{C}_r, \ell_\alpha \leq j\}$, the j 'th entry of C_r is then computed as:

$$C_r[j] = \frac{1}{|\mathcal{C}_r^j|} \sum_{\hat{P}_\alpha \in \mathcal{C}_r^j} \hat{P}_\alpha[j]$$

where $v[j]$ denotes the j 'th entry of vector v .

The clustering is initiated by selecting k profiles at random as the initial clusters, and then continually reassigning profiles and recomputing centroids for clusters. Having reassigned the profiles, the cluster error E_C is computed as the average distance between any profile and the centroid of its cluster. The algorithm iterates until the change in E_C is sufficiently small ($E_C \leq 10^{-6}$), or until a set number of maximum iterations (100) is reached.

Selecting the number of clusters k is an inherent problem in all unsupervised learning tasks. We use the so called *elbow heuristic* which relies on looking at how the error decreases with the number of cluster and pick at the "elbow" in the resulting curve. For each of the resulting clusters, we compute a few statistics and writing quality indicators, such as *noun and verb phrases* (the ratio between nouns/main verbs and sentences respectively) [5], and the *simple measure of Gobbledygook* (SMOG) grade [4] (which estimates the grade level required for understanding the text).

3. EXPERIMENTS AND RESULTS

The data consists of around 130K essays by approximately 10K students, with an average length of about 6K characters. The data set was cleaned by removing very short (≤ 400) and very long ($\geq 30,000$) texts, in order to get rid of invalid essays (blank hand-ins, garbled texts, etc.). Proper pronouns were substituted with placeholder tokens and the first 200 characters of each text were removed, in an effort to remove any data identifying the real author of the text, that could be picked up by the neural network and lead to overfitting. Finally, authors with less than 5 texts were removed.

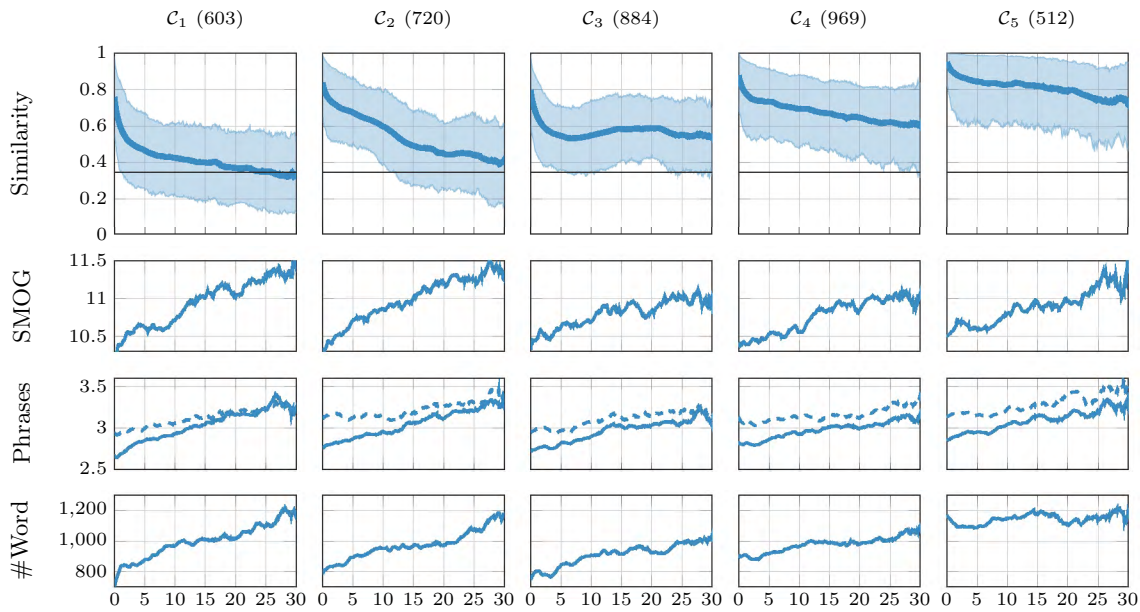


Figure 1: Plot of similarity curve, SMOG grade (incl. 90% intervals), noun/verb phrases (solid/dashed respectively) and average length (in words) for each cluster. The cluster size is given in parenthesis.

The resulting set is partitioned into two author disjoint sets: $T_{network}$ and $T_{analyze}$, used for training the network and performing the clustering respectively. $T_{network}$ is further split into author disjoint training and validation sets, T_{train} and T_{val} . The sizes of the sets are given in Table 1.

For training and evaluating the network, problem instances, consisting of pairs of texts, are constructed from T_{train} and T_{val} . A positive sample is created by selecting two texts from the same author, while a negative sample is created by sampling two texts from different authors. A 50:50 balanced data set is generated in this way². The number of instances (SIM) is shown in Table 1. The network is implemented using TensorFlow, and optimized for cross entropy using the Adam optimizer, using T_{val} for early stopping. For the clustering, each data point of $T_{analyze}$ simply consists of a single student and their texts, including meta data such as time of hand-in.

The profiles are constructed using $m = 2$. $k = 5$ is determined to be optimal using the elbow method and considering $k = 2, 3, \dots, 9$. Thus we obtain five clusters: C_1 , C_2 , C_3 , C_4 , and C_5 , with a cluster error of $E_C = 0.01407$. The curves representing the clusters (including cluster sizes and 90% distribution intervals) are shown in Figure 1³. The SMOG grade, the noun and verb phrases, and the average text length (in words) are also plotted. Furthermore, we sample two million random pairs of texts with random (different) authors, and compute the similarity for these samples, obtaining an average of 0.3470. This average is also plotted in Figure 1, while the similarity with respect to time is plotted as a heat map in Figure 2.

²We assume all claimed authors in the data to be the real authors; a few students may use ghostwriters or plagiarism, in which case some labels will be wrong.

³When visualizing and inspecting the clusters, we consider data until 30 months, as only few students are active longer.

4. ANALYSIS AND DISCUSSION

We start the analysis by considering the individual clusters found. When analyzing the clusters, three properties are important: the *initial value of the curve* describes the similarity between the second text of a student and their initial writing style, the *shape of the curve* describes the rate of change in writing style, and the *total change* describes how much the writing style has evolved. Furthermore, we will consider the mentioned indicators of writing style quality.

Across all clusters, the number of words written increases (with the exception of C_5), and the increase seems to be correlated with the corresponding decrease in similarity. Furthermore, students in all clusters appear to be improving with respect to the quality metrics on average. While positive, some clusters see a smaller increase than others, indicating that these clusters represents suboptimal development profiles.

C_1 and C_2 both represent students with a fairly large total change in writing style, and with a fairly high initial similarity. Both clusters also exhibit a large increase in the quality indicators (SMOG and phrases). The main difference is the rate of change of similarity; students in C_1 appear to change more rapidly in the first months before stagnation occurs, while students in C_2 appear to change more gradually. From a learning perspective, the development of students in C_2 seems superior. However, in terms of end result, it appears the writing quality of students in both clusters improves the same amount.

C_3 exhibits an interesting development pattern, in which the writing style appears to not only stagnate, but actually revert back, i.e. the similarity increases after the first year. This could be an indication of acquiring bad writing skills and 'unlearning' them again, or it could be an indication of learning new writing skills, but forgetting them again. In

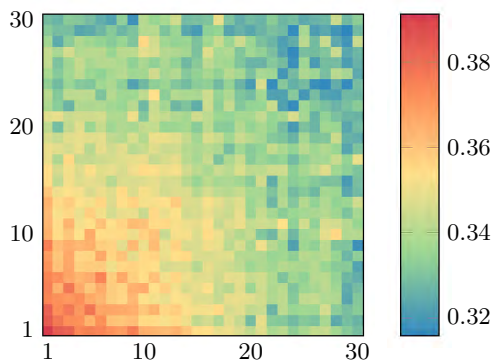


Figure 2: Heat map showing the average similarity between different authors, depending on how long time the two authors have been in high school.

any case, it seems the development is suboptimal. This is further supported by the smaller increase in quality indicators compared to C_1 and C_2 .

Considering the total development of students in C_4 , the result appears similar to C_3 . Students only see a limited total change in both similarity and quality indicators. However, compared to C_3 , students do not appear to revert back, but rather exhibit a slow development. Compared to clusters C_1 and C_2 , the development still appears suboptimal.

Finally, C_5 appears to be quite distinct from the other clusters, with a higher-than-average initial similarity and SMOG grade, and the smallest total change in similarity among the clusters. The high initial SMOG grade indicates that C_5 contains initially strong students (wrt. writing), while the small change in similarity and SMOG grade (compared to C_1 and C_2) indicates stagnant development; this could be an indication, that schools do not manage to properly encourage/teach students, who are initially strong.

In summary, C_1 and C_2 appear optimal, while C_3 and C_4 appear suboptimal. While the development of C_5 seems suboptimal, the students are initially strong, and thus not as much a point of concern, as in C_3 and C_4 . Looking at the number of cluster members, we see that C_3 and C_4 are the largest individually, indicating that quite a few students are exhibiting suboptimal writing style development. However, the majority of students are located in C_1 , C_2 and C_5 , indicating optimal (or at least not at-risk) development through high school.

Figure 2 explores the similarity between different students across time spent in high school. As mentioned, the average similarity between random students is 0.3470.

The plot shows students starting out similar in writing style and then becoming less similar as time passes. Surprisingly, a first year student and a third year student are equally or more similar in writing style on average, compared to two different students in their third year. This might be explained by an initially small space of possible writing styles, which grows as students are educated. One might expect some writing styles would diminish or even disappear, but apparently, more new and diverse writing styles develop. While education is sometimes accused of destroying individuality and/or creativity, these findings indicate

the opposite, at least in regards to writing style.

5. CONCLUSIONS

We presented a method for analyzing writing style development, by training a Siamese neural network to compute a writing style similarity measure, and using this measure to construct and cluster writing style development profiles.

An analysis was performed on a data set consisting of Danish essays from Danish high school students. Five clusters were found. Using noun/verb phrases and SMOG grade, two were determined to exhibit suboptimal development, indicating students possibly at risk. Furthermore, we saw how students become less alike during high school, as their writing styles diverge and become more individual.

6. ACKNOWLEDGMENTS

The work is supported by the Innovation Fund Denmark through the Danish Center for Big Data Analytics Driven Innovation (DABAI).

7. REFERENCES

- [1] Fazli Can and Jon M. Patton. Change of writing style with time. *Computers and the Humanities*, 38(1):61–82, Feb 2004.
- [2] Niels Dalum Hansen, Christina Lioma, Birger Larsen, and Stephen Alstrup. Temporal context for authorship attribution: a study of Danish secondary schools. In *Multidisciplinary information retrieval*, pages 22–40. Springer, 2014.
- [3] Annie Louis and Ani Nenkova. What makes writing great? first experiments on article quality prediction in the science journalism domain. *Transactions of the Association for Computational Linguistics*, 1:341–352, 2013.
- [4] Harry G. McLaughlin. SMOG grading - a new readability formula. *Journal of Reading*, pages 639–646, May 1969.
- [5] Emily Pitler and Ani Nenkova. Revisiting readability: A unified framework for predicting text quality. In *EMNLP 2008*, 2008.
- [6] Chen Qian, Tianchang He, and Rao Zhang. Deep learning based authorship identification. <https://www.semanticscholar.org/paper/Deep-Learning-based-Authorship-Identification-Qian-He/ab0ebe094ec0a44fb0013d640b344d8cfd7adc81>, 2018. Accessed April 2019.
- [7] V. Spandel. *Creating Writers: Through 6-trait Writing Assessment and Instruction*. Longman, 2001.
- [8] Efsthios Stamatatos. A survey of modern authorship attribution methods. *J. Am. Soc. Inf. Sci. Technol.*, 60(3):538–556, March 2009.
- [9] Magnus Stavngaard, August Sørensen, Stephan Lorenzen, Niklas Hjuler, and Stephen Alstrup. Detecting Ghostwriters in High Schools. In *27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2019.
- [10] Yuting Yang, Juan Cao, Mingyan Lu, Jintao Li, and Chia-Wen Lin. How to write high-quality news on social network? predicting news quality by mining writing style. *CoRR*, abs/1902.00750, 2019.

Learning Feature Analysis for Quality Improvement of Web-Based Teaching Materials Using Mouse Cursor Tracking

Mizuho Ikeda

School of Science and Technology, Kwansei Gakuin University
2-1, Gakuen, Sanda, Hyogo, 669-1337 Japan
gpikeda@kwansei.ac.jp

ABSTRACT

In blended learning, it is meaningful to obtain data on and understand the nature of the usage of web-based teaching materials by individual students and by whole classes in progress. For this purpose, a system called MCTA1, which can detect the behavior of students' mouse cursors on web-based teaching materials in real time and visualize the analysis results, was developed. MCTA1 provides functions that collect specific granularity as subsections and visualize the result of analysis in real time. Through the application of this system in real classes, particularly in programming practice lessons, it became clear that the behavior of the mouse cursor was an effective source of information for determining how individual students and whole classes actually use the web-based teaching materials. This paper describes an outline of how the MCTA1 system works. The analysis of students' usage under the actual conditions of each subsection is shown using a Lorenz curve and Gini coefficient. Additionally, the result of the analysis of the web browsing order route of each subsection is shown using Ward's clustering method.

Keywords

Mouse Cursor Tracking, Learning Feature, Quality Improvement, Lorenz Curve

1. INTRODUCTION

With the spread of education via electronic mediums, under various specialized fields, studies on the learning style [1], the effects of the education method utilized [2], and the operability of these systems have been actively conducted.

In order to improve the quality of the content of the web-based teaching materials, it is important to analyze exactly how the students use these teaching materials in their classes. In blended-type education, it is necessary to grasp information on the usage of web-based teaching materials under real

conditions and determine the students' usage of these materials during actual classes in progress. In this situation, collecting the mouse cursor tracks is considered one of the effective means for determining their actual usage of the web-based teaching materials. Therefore, a system called MCTA1, which can detect the behavior of students' mouse cursors on web-based teaching materials in real time and visualize the analysis results, was developed. Since the system does not hang from the load that normally results from using special instruments on students, it can collect more precise data on the students' natural state. The quality of web-based teaching materials can be considered to consist of three components (curriculum composition, structure of web-based teaching materials, and lesson planning). The structured design of each quality component was drafted, and web-based teaching materials were constructed [3].

These materials have been used in blended learning sessions that use the e-learning system. The average number of members in a class is 25. Each session runs for 90 min, and each subject is performed over 14 sessions. The students who take the class belong to a department of liberal arts and are assumed not to have much knowledge of mathematical science and information science.

In this paper, a system that can detect the behavior of a student's mouse cursor is introduced. The action data from each subsection of the web-based teaching materials pages, which have been developed and implemented, were also discussed. Furthermore, the effect of analyzing the students' unique behavior via the segmentation granularity of the learning log is also described.

2. RELATED STUDIES

When LA research is roughly classified, it consists of the evaluation of the collection processes of (1) study data, (2) analysis, and (3) analysis result. From the point of view of detecting student behavior, learning the history data from the learning management system (LMS) as a collection of data in (1) above, the discovery of digital data, such as mouse cursor behavior [4], facial actions [5], eye tracking [6] has been studied using surveys of self-reported data. Moreover, in order to estimate it as an analysis (2) of the above, or the above-mentioned analysis result (3), various visualization techniques have been proposed. In a research on the learning style and actions of a student, most of the experiments applied limited experimentation and evaluation.

Mizuho Ikeda "Learning Feature Analysis for Quality Improvement of Web-Based Teaching Materials Using Mouse Cursor Tracking" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 576 - 579

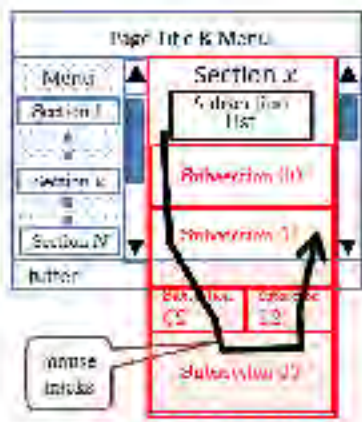


Figure 1: Web page composition and mouse cursor trackings.

These experiments were also performed in a restricted environment. There are few studies in real classes with conditions of continuous evaluation. Many analyses on the behavior of individual students have been conducted, but these studies often included only a few students and were limited by a lack of visualizations of group behavior.

When browsing a web page, the searcher tends to follow the mouse cursor with their eyes in order to easily click on the link of interest on the web page [7]. Based on this, the method of detecting the action of a mouse cursor is employed by this research. The system then detects whether the cursor would be in the position of a particular subsection based on the coordinate information built from the actions of many mouse cursors. The design and mounting were performed such that the automatic collection of continuous data could be realized, and a complete analysis of a student's features was conducted.

Section 3 describes the requirements for the functional enhancement of learning log collection, the system design of the action detection of a mouse cursor, and the mounting. Section 4 shows the system applied to a real classroom session and the effect of the feature analysis based on the learning log, subdivided according to the mouse cursor behavior data.

3. MECHANISM OF MCTA1

The web-based teaching materials used in real classes some pages consists of a menu, which contains the links to each section, and a section that is composed of several subsections.

At the top of each section, the subsection link is shown. Each indication range next to the subsection list is distinguished with a subsection number xy ($x = 0 \sim 9$, $y = 0 \sim 9$). Here, x shows the right and left domain, and y shows the top and bottom domain. Whenever the mouse cursor moves from one subsection to another, the numbers of the subsections that the cursor moves over in the process are sequentially collected automatically. The mechanism that collects both the subsection numbers and the time that a mouse cursor has been in existence has been developed.

Figure 1 is an example where the web-based teaching materials are comprised of five sections, and Subsection00 and a part of Subsection01 are displayed first. The shadowed area in the middle of Subsection01, Subsection02, Subsection12, and Subsection03 is shown. The arrow in Figure 1 is a trace of the mouse cursor tracks moved [Subsection00] → [Subsection01] → [Subsection02] → [Subsection03] → [Subsection12] → [Subsection01]. The y-axis direction is collected as the data "012321," and the x-axis direction is collected as "000010." At the same time, the time spent in each subsection is collected automatically.

The analysis and its visualization in real time are enabled by detecting the coordinate information, showing the action of a mouse cursor per subsection. The subsection of the middle position is only passed if the scroll bar is used and the trace data of the mouse can be collected sequentially. In addition, even if multiple windows such as when multiple tabs or separate windows are open are used alternately, the data can still be accurately collected.

4. VISUALIZING LEARNING FEATURE

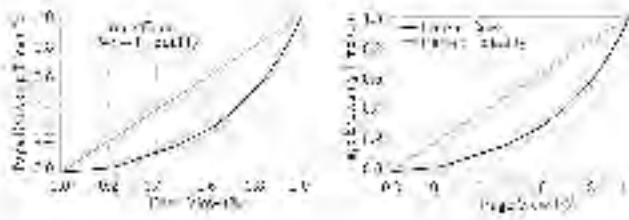
By using mouse cursor tracking data, the distribution difference of the browsing subsections is validated. The data also allows checking of whether the browsing distribution of the web-based teaching materials does not result in a difference when the unique features of each student are analyzed. Here, a Lorenz curve and the Gini coefficient are used to evaluate the inequality of page browsing as a statistical value. In fact, these are commonly used statistics to express inequality, such as with income in economic science. As the Gini coefficient approaches 0, the Lorenz curve approaches the complete equality line, and the equability becomes high. As the Gini coefficient increases, the Lorenz curve keeps away from the complete equality line, income disparity decreases, and the equability decreases. The Lorenz curve, the complete equality line, and the Gini coefficient were used to analyze the reference time of each subsection of the web-based teaching materials.

The possibility of referring to each subsection equally becomes high, so that the Gini coefficient calculated from the accumulation reference time of web-based teaching materials approaches 0. In a blended-type educational setup, the students are expected to follow the footsteps to the subsection which the teacher is explaining and likely to refer to the said subsection. Therefore, the closer the Gini coefficient is to 0, the higher the possibility that the students are wandering around to another subsection. The higher the Gini coefficient, the lower the equability of the subsection reference. One subsection is more likely to be intensively referenced.

To determine the students' usage situation, clustering was then performed on the patterns of the browsing orders of the subsections. Ward's hierarchical agglomerative clustering method was adopted because it tended to be easy to clearly classify into a cluster.

4.1 Visualization using Lorenz curve

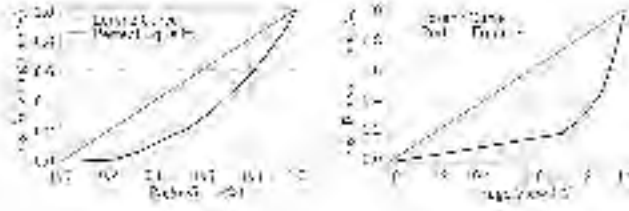
A Lorenz curve, represented by the values along the y-axis, is the cumulative relative frequency of the browsing time



(a) Level 2.

(b) Level 5.

Figure 2: Lorenz Curve (Level)



(a) Session 2.

(b) Session 13.

Figure 3: Lorenz Curve (Session)

of subsections, with its x-axis being the cumulative relative frequency of the browsing number of subsections used to visualize the inequality of the browsing time distribution. First, the browsing distribution according to each student's level, i.e., degree of subject achievement, was compared.

Based on achievement in the applied subject, the five steps for gauging the student's levels were evaluated, and a group division was carried out. The level is a five-step evaluation based on the number of tasks achieved for 38 subjects.

The Lorenz curve for level 2 is shown in Figure 2(a), and for level 5 is shown in Figure 2(b). Here, level 5 implies the level with the highest degree of subject achievement. The Gini coefficient was 0.45 at level 2 and 0.44 at level 5; therefore, the difference was not so apparent. Next the browsing distribution according to each lesson was compared. The browsing time distributions of the contents of the web-based teaching materials for all students of the 2nd session and the 13th session are indicated in Figure 3(a) and Figure 3(b), respectively. The Gini coefficient for the 2nd session was 0.41 and for the 13th session was 0.60. Compared with the first session time, the Gini coefficient tends to increase. This demonstrates that the inequality of the browsing distribution becomes high. Therefore, it is shown that the possibility of perusing one's subsection is high toward the second half of a lesson.

4.2 Time series display of Gini coefficient

The time series transition of the Gini coefficient in Figure 5 is a graph in which the session extends in the x-axis direction and Gini coefficient is set to the y-axis direction. The lower the Gini coefficient, the higher the browsing homogeneity. This shows that various subsections are browsed uniformly. The higher the Gini coefficient, the higher the uniformity. This shows that a few subsections are browsed intensively.

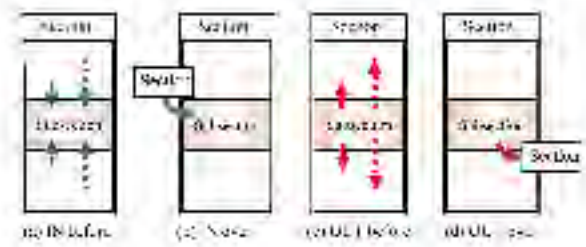


Figure 4: Patterns of browsing order.

The Gini coefficients of students from all levels except level 1 do not fluctuate very much. Particularly during the second half of the term (Sessions 8–14), the Gini coefficient of the levels 4 and 5 students maintain a relatively static state in a location for which it is high. In the first half of the term, the students browse not only the same subsections as the teacher, but other subsections as well. Therefore, the browsing distribution can encounter an inequality fluctuation for a short while. It later turns out that as the second half of the term progresses, the students become more concentrated in one subsection while browsing.

The level 2 students follow the teacher's browsing transition to the subsection, similar to what the level 5 students do in the first half of the term (Sessions 1–7). However, after Session 8, the homogeneity of the students' subsection browsing becomes much higher, and it seems to become clear that they browse various subsections, including past subsections, in addition to the subsections that the teacher browses.

In addition, the figure representing the time series of the Gini coefficient can be shown in real time. The teacher is able to confirm the non-homogeneity of the browsing distribution and adjust the speed of the teaching progress by returning to past subsections and lectures. Furthermore, it becomes useful in thinking about web-based material contents and the relocation of subsections.

4.3 Browsing order and route

Next, to clarify the students' features, representing their browsing behaviors, their browsing routes to each subsection were analyzed.

4.3.1 Patterns of browsing order

The students' mouse cursor tracks to each subsection are classified into the following four patterns. The pattern of where the mouse cursor came from and where it was moved to can be classified into the following four types (see Figure 4):

- (a) **IN before** : Move from one subsection to the target subsection in the same section
- (b) **IN over** : Move from one section to the target subsection
- (c) **OUT before** : Move from the target subsection to another subsection in the same section
- (d) **OUT over** : Move from the target subsection to another section

4.3.2 The graph of each cluster feature

According to these attributes, clustering is performed and a dendrogram is constructed by Ward's hierarchical agglom-

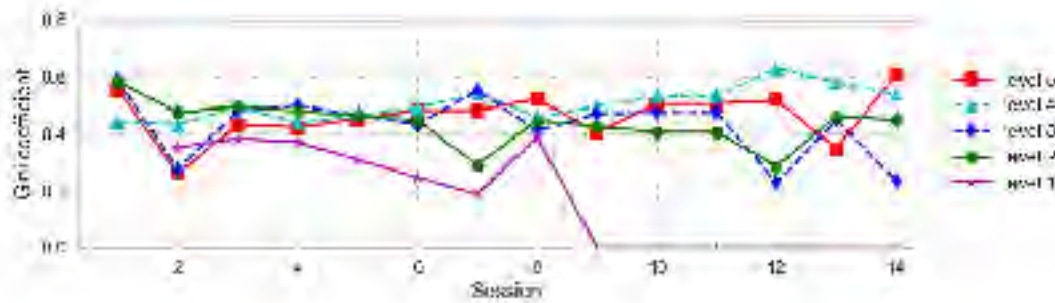


Figure 5: Transition of the Gini coefficient of each session (each student level).

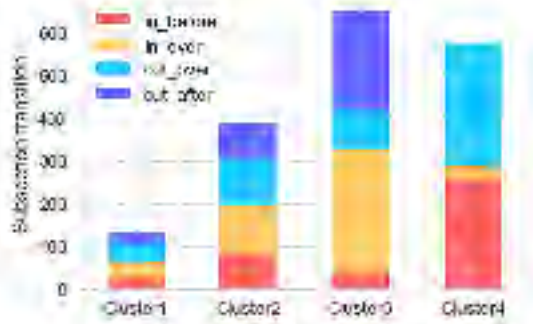


Figure 6: Characteristic of the clustering result.

erative clustering method. In Figure 6, when height is set to 750, number of clusters is set to four. To confirm the four clusters' features, stacked graphs were created. According to the list of subsections of each cluster, the following features were clarified:

Cluster1 : Students browse repeatedly through various subsections without following the composition order of the subsection. It occurs in the subsection concerned with the beginners' tasks in the first half of the term.

Cluster2 : Like in "Cluster1," students browse repeatedly through various subsections without following the composition order for the subsection. However, the subsection is browsed more times than in "Cluster1." It is also occurring in the subsection concerned with basic tasks in the first half of the term.

Cluster3 : Students transferred from the subsection without following the order of subsection composition and transferred to the next subsection according to a composition order. It is occurring in the subsection concerned with the high difficulty level tasks in the second half of the term.

Cluster4 : Students transferred from the subsection according to a composition order and transferred to the other subsection without following the order of subsection composition. Like in "Cluster3," it is occurring in the subsection concerned with the high difficulty level tasks in the second half of the term.

Classifying subsections seems to help with the reconstruction of the web-based teaching materials so that students' understanding can be increased in stages.

5. DISCUSSION

Visualization using Lorenz curve, the time series transition of the Gini coefficient and the graphs of the clustering meth-

ods can be leveraged for information to support the class as the lessons progress. In addition, it was found that the results of the clustering of students' transition route is useful information for reviewing reconstruction, rearrangement, and speed adjustment of class progress, as the quality model of web-based teaching material.

6. CONCLUSION AND FUTURE WORK

A system to collect mouse cursor behavior automatically has been designed and developed. In addition, it has been applied to an actual class. Learning logs have also been collected. Furthermore, a system that can visualize in real time the progress of a lesson using the learning log has been developed. It was found that it is possible to clearly capture a student's learning activities. A learning log of the subsection units has been accumulated every semester since the 2015. A quality evaluation model of web-based teaching materials needs to be established by analyzing additional learner features (e.g., the students' learning style) and applying them to a mathematical model.

7. REFERENCES

- [1] T. A. Litzinger, S. Lee, J. C. Wise, and Richard M. Felder. A Psychometric Study of the Index of Learning Styles. In *Journal of Engineering Education*. 96(4), pages 309–319, 2007.
- [2] N. Bos, B. Meijer and S. Brand-Gruwel. A network analysis of blended learning. In *EARLI SIG4*, 2016.
- [3] M. Ikeda. A Development and Implementation Model for Evaluating Web-Based Teaching Materials. In *Research Report of JSET Conferences*. 2012(4), 49–54, 2012.
- [4] J. Schneider, M. Weinmann, J. vom Brocke, and C. Schneider. Identifying Preferences Through Mouse Cursor Movements Preliminary Evidence. In *ECIS2017*, pages 2546–2556, 2017.
- [5] A. K. Vail et al. The Affective Impact of Tutor Questions: Predicting Frustration and Engagement. In *EDM2016*, pages 247–254, 2016.
- [6] S. Hutt, C. Mills, S. White, P. J. Donnelly, and S. K. D'Mello. The Eyes Have It. : Gaze-based Detection of Mind Wandering during Learning with an Intelligent Tutoring System. In *EDM2016*, pages 86–93, 2016.
- [7] P. B. Kantor, E. Boros, B. Melamed, V. Menkov, B. Shapira, and D. J. Neu. Capturing Human Intelligence in the Net. In *ACM 08/2000*, Vol.43, No.8, pages 112–115, 2000.

It's a Match! Reciprocal Recommender System for Graduating Students and Jobs

Anik Jacobsen and Gerasimos Spanakis

Maastricht University

Maastricht, 6200MD, Netherlands

anik.jacobsen@alumni.maastrichtuniversity.nl, jerry.spanakis@maastrichtuniversity.nl

ABSTRACT

Finding the perfect job that takes into account someone's skills and ambitions is an overwhelming challenge for many freshly graduated students. At the same time, companies struggle to hire employees fulfilling their requirements. Ideally, a successful match of students and jobs includes the preferences of both sides. This paper proposes a reciprocal recommender system matching graduating students and job offers. To construct a common representation space for the items of the recommendation, the course descriptions from the curriculum are used as a linking factor. Further, Latent Dirichlet Allocation (LDA) is used to extract topics from the course and job descriptions, forming the latent representation space. Next to providing job recommendations, curricula gaps can be discovered and thus the students can be better prepared for a future career. The algorithm is tested on data from a Data Science bachelor programme. Results show that a reciprocal matching of graduating students and jobs generates recommendations with precision and recall up to 0.7. The approach yields promising results for such system to be implemented as a job recommender system on university level or as a stand-alone system for graduating students.

Keywords

reciprocal recommender systems, job matching, Latent Dirichlet Allocation, discovering curricular gaps

1. INTRODUCTION

In recent years, most of the recruiting process has been shifted to online job portals and professional social networks, such as LinkedIn. Many graduating students feel overwhelmed by the available quantity and struggle to find a position that fits the curriculum of their degree, as well as the skills and interests developed during their studies. Moreover, employers are reluctant to employ freshly graduated students since they lack knowledge about the curriculum content and the skill sets of possible candidates. Many

degrees fail to address the requirements of job positions and often have gaps in the curricular, which make it difficult for students to start their career.

Job matching is a complex problem, as the employee candidate should fulfil the employer's requirements and the job should realise the candidate's ambitions. The situation is comparable to a dating scenario. The aim is to match students and jobs reciprocally, taking into account preferences from both sides. To do so, this paper proposes a reciprocal recommender system, matching graduating students and job offers based on the students' study curricula. To construct a common latent representation space, a probabilistic topics model is used, extracting topic distributions from the curriculum course and job descriptions.

The remainder of the paper is organised as follows: Section 2 presents related work in the domain of reciprocal recommenders and job recommender systems. In Section 3, the data and the proposed method used for generating recommendations are presented. After, in Section 4, the experiments are discussed and evaluated. Finally, Section 5 concludes the paper and presents future work.

2. RELATED WORK

Reciprocal recommenders are a class of recommender systems that consider the preferences of both items which are part of the recommendation, a match occurs when both sides benefit from a recommendation. The most significant research in the field has been done in social matching, especially online dating[1, 8] but also study buddy matching[9].

The process of recruitment has been shifted to an online environment. Thus, there is a high demand for sophisticated recommender systems. Yao et al. [7] propose a hybrid recommender system, taking into account user profiles and interactions. Their recommendations are based on a directed, multi-relational, weighted graph, whose structure is provided by the interactive features of a job seeking website. iHR+ is a mobile reciprocal job recommender system [10], which divides the information of job seekers into two categories: self-description and preference features. Recommendations are generated using the cross-similarity among these features. Similar to the approach presented in this paper, Ding et al. [5] focus their reciprocal job recommender system on current graduating students. Their system takes historical student information, as well as the preference of employers and students into account. Using the grading in-

Anik Jacobsen and Gerasimos Spanakis "It's a Match! Reciprocal Recommender System for Graduating Students and Jobs" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 580 - 583

formation of former and current graduates, they determine candidates (in terms of employers and students) similar to comparable students.

In the domain of generating recommendations using probabilistic topic models, Apaza et al. [2] propose a content-based recommender system for online courses. They use LDA to derive topic distributions as descriptive feature vectors for the courses. Combining this with the students' historic grading information, they are recommending courses to students with weaknesses.

This paper focuses on matching graduating students with jobs. The common latent representation space is formed by the course and job descriptions. The students' grading information is used as a preference and strength indicator. Thus, there is no intermediate step explicitly stating the preferences for both sides of the recommendation. Developing a job recommender system based on topics derived from course and job descriptions is new in the domain of reciprocal recommenders.

3. PROPOSED METHOD

3.1 Data

The data used for the recommendation has to describe three items: the jobs, the courses, and the students.

The job offers were scraped from indeed.com, to obtain realistic data. indeed.com is a job search engine. It obtains its entries from multiple sources, such as company websites, postings created on indeed.com itself or job boards. The target positions were derived from alumni data. The keywords "junior" or "entry-level" made it possible to identify job offers suitable for freshly graduated bachelor students. Further, the scraping was limited to job offers in the English language, to ensure comparability with the course descriptions. Eventually, 103 job offers were collected, suitable for someone with a Bachelor degree in Data Science.

The programme's student guide provides a course description for each course in the curriculum. They are phrased according to the Dublin Descriptors, which provide information about the content and structure of a course, as well as the competences acquired in a course.

Overall 72 student grading data points were used. It was obtained from two sources: anonymised student information from 2011 to 2015 (44 students) and grading information from 28 current students in their final year. All the grades are in the range from 1 to 10.

3.2 Recommendation Algorithm Description

An overview of the recommendation algorithm is displayed in Figure 1. In summary, first, the student grades are normalised. Then, a Latent Dirichlet Allocation (LDA) model is trained and applied on a corpus consisting of the job postings and course descriptions. The resulting topic distribution matrices are used to compute a similarity matrix of the jobs and courses. Combining the similarity matrix and the normalised student grades, a score is computed, allowing to generate ranked recommendation lists for students (recommending jobs) and jobs (recommending students).

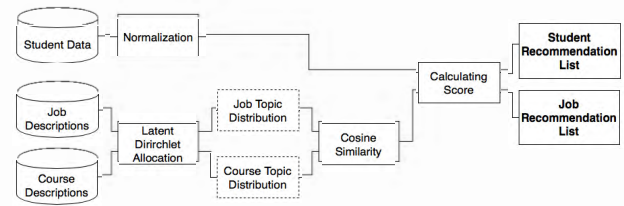


Figure 1: Pipeline of the recommendation algorithm

3.2.1 Normalising Student Data

To obtain successful matches for all students, the grading information needs to be normalised. Examiners grade in different ways. This grading bias is removed, by subtracting the mean in the course dimension from each datapoint. The aim of using the grades is identifying strengths and interests of the students. To obtain a "strength profile" per student, the mean in the student dimension is subtracted from each datapoint. Lastly, the grades which have a missing value are replaced with zeros. Students did not attend the course and thus they should have a zero weight for the final score.

3.2.2 Applying LDA and Constructing Similarity Matrix between Courses and Jobs

Topic models are statistical models for discovering abstract topics that appear in a corpus of documents. In the application, Latent Dirichlet Allocation (LDA) is used. It was introduced by Blei et al. in 2003 [4]. The model is based on the assumption that each document is a combination of topics and each topic is a probability distribution over words [3]. From a set of documents and a fixed number of topics, LDA returns a set of words associated with each topic with a probability. Applying the trained model to a document gives a probability distribution for the topics.

To compare the course descriptions and job postings, feature descriptors for the text documents need to be constructed. Assuming that each course description and job offer includes a certain set of topics and there is a set of relevant words describing the topics, a probabilistic topic model (LDA in this case) can be used to describe the topics of the documents. The model is first trained on the corpus consisting of the course and job descriptions. 12 is chosen as the number of topics for the algorithm because content analysis of the topics showed this was the optimal number.

The result of applying the trained LDA model to the corpus is a feature vector per document, expressing the probability of each topic. The vectors are combined in two matrices, one describing the courses and one describing the jobs. The cosine similarity is used as similarity measure. The resulting similarity matrix gives the similarity for each course and job respectively.

3.2.3 Calculating Score and Ranking Recommendations

The final score for each student and job combination is a matrix multiplication. The courses are the linking factor between the students and jobs. Thus, the score is calculated by multiplying the normalised grades per course of each student with the similarity matrix of courses and jobs. This is formalised by Equation 1.

$$score_{s,j} = \sum_{c=1}^M grade_{s,c} * sim_{j,c} \quad (1)$$

where s is the student, j is the job and c is the course. M is the number of courses. As the final recommendations, the jobs with the top k scores for each student (ranked), and the students with the top k scores for each job (ranked) are recommended.

4. EXPERIMENTS AND RESULTS

An appropriate evaluation of the recommender system requires the deployment of the system and feedback from students and recruiters. In this case, it is only possible for a part of the students. Prabhakar et al. [9] propose measures to evaluate a reciprocal recommender system using the available data. Next to evaluating the experiments using the proposed measures, recommendations generated by deploying the algorithm were sent to the 28 current final year students to obtain qualitative feedback.

4.1 Evaluation Metrics

A successful reciprocal recommendation includes the preferences of both sides. Thus, it can be said that “Student y is a successful (reciprocal) recommendation for job x (out of the k -total), if and only if, job x is also in the recommendation list of student y .”. The same holds for the job recommendations to students. Based on this logic and the precision and recall measure modified by [9] we can formalise precision and recall as follows:

$$P_x = \frac{N_x}{k}, R_x = \frac{N_x}{N_{*x}} \quad (2)$$

where P_x is the precision for student/job x , R_x is the recall for student/job x , N_x is the number of successful recommendations for student/job x (as defined before), k is the total number of recommendations generated and N_{*x} is the number of jobs/students that have x in their recommendation list. The total precision and recall of all recommendations can be computed using the following formulae:

$$P = \frac{1}{M} \sum_{i=1}^M P_i, R = \frac{1}{M} \sum_{i=1}^M R_i \quad (3)$$

where P_i and R_i are the precision and recall respectively for student/job i (as defined previously) and M is the total number of students/jobs.

The results of the recommendation system are two recommendation lists, one recommending jobs to students and one recommending students to jobs. Precision and recall are first calculated separately for the students (P_S, R_S) and the jobs (P_J, R_J), thus per recommendation list. For the final measure, the mean is computed.

Another measure proposed by Prabhakar et al. [9] is the Discounted Cumulative Gain (DCG). Originally, it has been developed for the field of information retrieval to measure the quality of rankings [6]. It was adapted by Prabhakar et al. to measure rank alignment or reciprocity [9]. A perfect rank alignment and thus an indicator for a good reciprocal system is described by: “For all students/jobs i , present at position j in the recommendation list of job/student u , if u is also present at the same position j in the recommendation list of i ”. Dividing it by the number of successful recommendations ensures that all values are in the range $[0, 1]$. If there are no successful recommendation for student/job u , the gain is 0. The formula is defined as:

Figure 2: Precision graph, including the standard deviation.

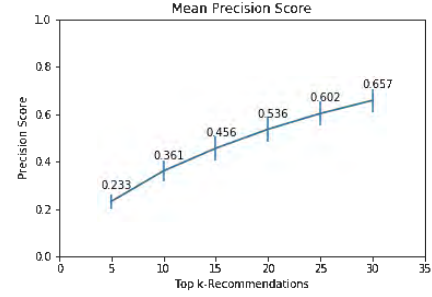
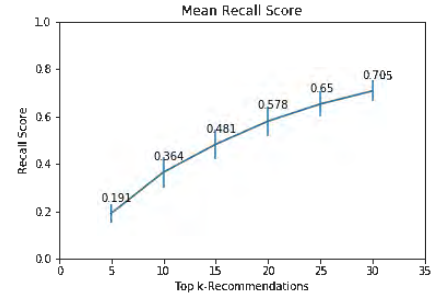


Figure 3: Recall graph, including the standard deviation.



$$DCG = \frac{1}{M} \sum_{u=1}^M \frac{\sum_{j=1}^k g_{uij}}{S} \quad (4)$$

where M is the number of students/jobs, S is the number of successful recommendations, j is the rank and g_{uij} is the gain of job/student i (at position j) for student/job u . The gain is given by:

$$g_{ui} = \frac{1}{1 + |diff_{ui}|} \quad (5)$$

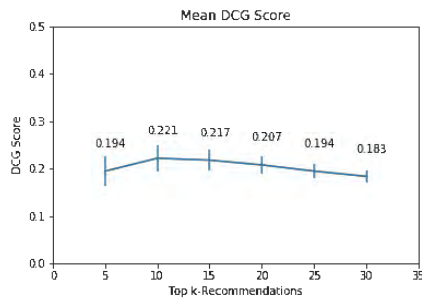
where $diff_{ui}$ is the difference in ranks of student i in the recommendation list of job u and job u in the recommendation list of student i . The same holds for job i and student u . Thus, the gain is 1 when the rankings between a student and a job are the same; otherwise, the gain is discounted. As before, the DCG is calculated separately for students (DCG_S) and jobs (DCG_J). Then, the mean is computed.

4.2 Quantitative Results

The recommender system is evaluated on the data described in section 3.1. The recommendations are ranked according to their score and the top-k [5,10,15,20,25,30] jobs/students are recommended. Because of the LDA topic model, there is some randomness involved in the process. 50 runs of the experiments are conducted and precision, recall and DCG scores are recorded. The results are averaged across the runs, for each top-k recommendations. The precision, recall and DCG graphs are shown in Figures 2, 3 and 4. The standard deviation is displayed as a stability indicator.

Precision and recall increase with the number of recommendations given. Giving more recommendations increases the probability of successful recommendations. The low variance shows that the results are reliable. Reaching a precision score of 0.657 and a recall score of 0.705 with $k=30$ recommendations indicates that most matches generated by the system are successful.

Figure 4: DCG graph, including the standard deviation.



The decreasing value of the DCG with recommendations given can be explained by the increasing difference in ranks in the recommendation list. The peak at $k=10$ (DCG score 0.221) indicates that a number of 10 recommendation is optimal to provide students and jobs with the most rank aligned recommendations. In the case of a job recommender system, a successful recommendation is more important than rank alignment. Thus, precision and recall scores provide a better insight into the quality of the recommender system. Overall, the precision and recall measures show that the reciprocal approach generates successful recommendations, matching students and jobs effectively.

4.3 Qualitative Results

The 28 last year students who gave access to their data were provided with their top 10 recommendations and were asked to re-rank them. To analyse the results, the DCG was used, as a measure of rank alignment between the original and re-ranked rankings. The DCG score for comparing the original and re-ranked entries with each other is 0.472. Since all items are represented in both lists, the DCG score is an average of the gain of all recommendations for all participants. Thus, a DCG score of 0.472 indicates that on average, each posting was re-ranked 1 or 2 rankings below or above its original ranking (since $\frac{1}{2} > 0.472 > \frac{1}{3}$). Some change in the ranking was to be expected because some of the job offers were similar to each other. The result supports that the order of the recommendations was appropriate.

The students were also asked to give qualitative feedback on their recommendations. The overall feedback received from the students was positive. Most reported that they received interesting input for future career perspectives. However, one drawback revealed by the feedback is the limited possibility of expressing interests and ambitions. Incorporating this information in the recommendation algorithm is a future direction.

5. CONCLUSION AND FUTURE WORK

This paper proposes an algorithm that matches graduating students and jobs reciprocally, simplifying the overwhelming job search for graduating students and allowing employers to recruit graduated students more effectively. It was shown that the system can be deployed to generate successful employment couples. The items of the recommendation are solely represented by the academic information from the students and the descriptions of the jobs. Courses are used as a linking factor, a common representation space for academic courses and jobs was established using LDA. Finally, quan-

titative results, based on measures introduced by Prabhakar et al. [9], show that the reciprocal approach is an appropriate way to generate successful matches. The feedback from students in their final year showed that the results could be used for career inspiration, however, a way to express preferences is desired.

The work presented in this paper provides many opportunities for further experimentation and improvement. Course preferences from the student side could be included to recommend jobs that better fit the ambitions and interests of the students. Moreover, feedback from the employer perspective could lead to new insights on how to enhance the algorithm to better match the recruiters' needs. The results could be used to discover curricular gaps in the study programmes. Some job descriptions might include topics which are not addressed by the courses. Tackling these issues would lead to students better prepared for the business world and its requirements. The implementation of the algorithm was limited to one programme. More (real-life) experiments would be useful to evaluate further and improve the system. Long term deployment of the algorithm within a suitable platform would lead to advanced conclusions.

6. REFERENCES

- [1] J. Akehurst, I. Koprinska, K. Yacef, L. A. S. Pizzato, J. Kay, and T. Rej. Ccr-a content-collaborative reciprocal recommender for online dating. In *IJCAI*, pages 2199–2204, 2011.
- [2] R. G. Apaza, E. V. Cervantes, L. C. Quispe, and J. O. Luna. Online courses recommendation based on lda. In *SIMBig*, pages 42–48, 2014.
- [3] D. M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.
- [4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
- [5] Y. Ding, Y. Zhang, L. Li, W. Xu, and H. Wang. A reciprocal recommender system for graduates' recruitment. In *Information Technology in Medicine and Education (ITME), 2016 8th International Conference on*, pages 394–398. IEEE, 2016.
- [6] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.
- [7] Y. Lu, S. El Helou, and D. Gillet. A recommender system for job seeking and recruiting website. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 963–966. ACM, 2013.
- [8] L. Pizzato, T. Rej, T. Chung, I. Koprinska, K. Yacef, and J. Kay. Reciprocal recommender system for online dating. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 353–354. ACM, 2010.
- [9] S. Prabhakar, G. Spanakis, and O. Zaiane. Reciprocal recommender system for learners in massive open online courses (moocs). In *International Conference on Web-Based Learning*, pages 157–167. Springer, 2017.
- [10] H. Wenxing, C. Yiwei, Q. Jianwei, and H. Yin. ihr+: A mobile reciprocal job recommender system. In *Computer Science & Education (ICCSE), 2015 10th International Conference on*, pages 492–495. IEEE, 2015.

Supporting Minority Student Success by using Machine Learning to Identify At-Risk Students

J.D Jayaraman
New Jersey City University and
Teachers College, Columbia
University
Jersey City, New Jersey
+1 (201) 200 2297
jjayaraman@njcu.edu

Sue Gerber
New Jersey City University
Jersey City, New Jersey
+1 (201) 200 3042
sgerber@njcu.edu

Julian Garcia
New Jersey City University
Jersey City, New Jersey
+1 (201) 200 3463
jgarcia8@njcu.edu

ABSTRACT

Student retention is a major challenge at American universities with the average six year graduation rate hovering around 59%. Among minority students the graduation rate drops to 46% for Blacks and 55% for Hispanics. Low graduation rates not only impact the financial well-being of individuals but the economy as a whole. Thus, improving student retention, in particular, minority student retention, is of paramount importance at institutions of higher education. This paper describes a machine learning approach to predicting minority native and transfer student dropout using a dataset from a four year Hispanic serving institution in the north eastern region of the United States with a large percentage of minority students. The results of the study show that standard machine learning models can predict minority transfer student dropout with a high degree of accuracy of 97% and minority native student dropout with an accuracy of 81%. The features that were most important in predicting minority transfer student dropout were SAT scores, and college cumulative GPA, while high school GPA and college cumulative GPA were the top predictors for minority native student dropout. This study demonstrates that educational institutions can use cost effective off-the-shelf standard machine learning models to achieve a high degree of accuracy in predicting minority student dropout. The high prediction accuracy achieved helps in reliably identifying at-risk minority students and providing them with necessary interventions to support their academic success.

Keywords

Dropout prediction, Minority student retention, Machine learning models, At-risk students

1. INTRODUCTION

Student retention is a major challenge at American universities with the average 6 year graduation rate hovering around 59% [14]. Graduation rates vary with institutional selectivity [18]; the situation being particularly grave at institutions with open admission policies where the 6 year average graduation rate is at a meager 32% [14]. There is substantial variation in the graduation rates by race and ethnicity. African American students had the lowest six year graduation rate at 46% while Hispanic students'

graduation rate is at 55% [14]. Transfer student graduation rates are also low at 42% [14]. Thus, improving student retention, particularly for minorities, is of paramount importance at institutions of higher education.

A critical factor in increasing student retention is the ability to accurately identify at-risk students, so that relevant interventions can be provided. But, there is a paucity of literature focused on predicting minority student dropout and even less literature that considers native and transfer students separately, taking into account their individual characteristics. Minority students' graduation rates are among the lowest and they have different characteristics, needs and face different challenges in college. Hence analyzing and modeling their dropout rate separately is warranted. It would also be informative to model the transfer and native students separately as these students have different characteristics and different graduation rates. Such an analysis could be used to identify, target and customize interventions differently to minority transfer and native students and hence lend better support for their success in college.

This paper describes a machine learning approach to predicting minority college student dropout, treating native and transfer students separately. The objective of this paper is not to build an esoteric novel machine learning model hitherto not seen in the literature, instead, we aim to show the effectiveness of standard off-the-shelf machine learning models in predicting minority student dropout. To the best of the authors' knowledge this study is one of the first to employ machine learning techniques to build separate models to predict minority transfer student and native student dropout taking into account the unique characteristics of each. Thus, this study contributes to the literature by demonstrating that standard machine learning models can achieve a high degree of accuracy in predicting minority student dropout by taking into account the different characteristics of native and transfer students. This high prediction accuracy can then be used to reliably identify at-risk students and provide them with the necessary intervention to support their success. This study also contributes by giving readers an idea of which machine learning models work well in this domain, what data preprocessing is needed and what features have good predictive power. Further, this study contributes to student retention practice as it demonstrates that easy to implement and cost effective off-the-shelf machine learning models can achieve a high degree of accuracy in identifying minority students at risk of dropping out.

2. LITERATURE REVIEW

Research on student attrition has traditionally been based on surveying student cohorts and following them to assess drop out. These surveys contributed to the building of theoretical models of student retention, the most famous of them being the Tinto model

J.D Jayaraman, Sue Gerber and Julian Garcia "Supporting Minority Student Success by using Machine Learning to Identify At-Risk Students" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 584 - 587

[17]. These survey based research have been criticized for being too specific to an institution and hence not generalizable [6]. An alternative to survey based research is to use the data that most higher education institutions routinely collect about their students. This type of research based on institutional databases has been shown to be comparable to survey based research [4].

There are numerous factors that affect student retention. Tinto [17] highlights academic difficulty, adjustment problems, lack of clear academic goals, lack of commitment, inability to integrate with the college community, uncertainty, incongruence and isolation as factors involved in student dropout. Tinto's theory of student integration posits that past and current academic success are crucial factors in determining student attrition and many studies have found high school GPA and SAT scores to have a strong effect on student retention [18]. Declaration of major and number of credit hours taken during the first semester have been used as proxies for institutional and goal commitment.

Transferring from one educational institution to another also has an impact on retention rates. Factors affecting transfer student academic success and persistence include issues regarding institution support [9, 23], financial factors [7], student goals [11] and familial support [1, 10].

When it comes to minority students Moffat [13] found that SAT scores were not a strong predictor of student success for Black students. A key challenge to success minority students face in predominantly white institutions is a sense of alienation due to underrepresentation [15]. Hoffman [10] found that student satisfaction and success can be strongly linked to cocurricular involvement for minority students.

Research on using machine learning techniques to predict student attrition is still in its infancy. Delen [8] and Thammasiri [16] used several machine learning methods such as support vector machines and neural networks to model freshmen student attrition and found that support vector machines performed best, reaching a prediction accuracy close to 80%. Lauria, Baron, Devireddy, Sundararaju, and Jayaprakash [12] used demographic and course related data to show that support vector machines performed better than decision trees at predicting at-risk students.

This study uses many of the factors impacting student retention identified in the literature to build machine learning models to predict student attrition. Factors unique to transfer students, as identified in the literature, are used to build specific models to predict transfer student dropouts.

3. CONCEPTUAL FRAMEWORK

The study is broadly based on the models of student retention developed by several researchers, one of the earliest and popular being that of Tinto [41]. Tinto's model suggested that student success is determined by the degree of academic and social integration. Other popular models of student retention include Bean's student attrition model [2, 3] which takes the employee turnover approach suggesting that students dropout for similar reasons as employees leave an organization and the Cabrera, Nora & Castenada [6] model which integrates the Tinto and Bean models. Based on these models several studies have identified factors that impact student dropout. High school GPA, SAT scores, number of credit hours taken during the first semester, declaration of major, aid based on academic achievement and student loans are among the factors that are predictive of student dropout.

Our study attempted to collect data on various factors based on the

theoretical models and the predictive factors that have been identified based on them and use them as features in our machine learning models.

4. METHODOLOGY

4.1 Data

This study used five years (2011 – 2015) of minority student data from a regional Hispanic serving four year college in the north eastern region of the United States. The university is an urban university catering to a largely minority population (80% minorities). The university accepts a large number of transfer students from the local community colleges and other institutions. The overall four year graduation rate at the university was around 55%, with the graduation rate among transfer students at 61% and native students at 45%. Table 1 presents some descriptive statistics of the dataset.

Table 1: Descriptive Statistics

Native Students		Transfer Students	
Number of students	3897	Number of students	4703
Female	58%	Female	65%
Male	42%	Male	35%
Hispanic	55%	Hispanic	48%
African American	30%	African American	34%
Asian	11%	Asian	15%
Other race	4%	Other race	2%
Mean age	19	Mean age	28
Mean GPA	2.50	Mean GPA	2.99
Mean SAT Math	450	Mean SAT Math	413
Mean SAT English	430	Mean SAT English	390

We define a student to have dropped out if he/she does not enroll in the year following the last semester of enrollment. Based on this definition we constructed a binary indicator variable to indicate whether a student has dropped out or not. Both the transfer student and native student dataset had about 35% dropouts.

4.2 Features

Table 2 shows the features that were used in the machine learning models.

Table 2: Features

Features used for both native and transfer students
Age
Gender
Race
High School GPA
Gateway Math Status (gateway math course is required or not)
Gateway English Status (gateway English course is required or not)
Math placement (Has student completed the developmental math requirement)
English placement (Has student completed the developmental English requirement)
Cumulative GPA
Trend in the GPA over the semesters enrolled (Increasing, Decreasing, Stable)
SAT Math
SAT English
Difference between credits taken and credits earned
Community involvement (Student belongs to clubs and other student organizations or not)

Has student declared major (yes or no)
Additional features used only for transfer students
Difference between number of college credits applied for transfer and accepted for transfer.
Marital status
Highest degree earned prior to transfer

The features we used broadly fell into the following categories: student achievement, performance and progress, community engagement and demographics. We engineered several of the features from the raw data. From the raw data on GPA we extracted a feature indicating the trend in the GPA. From the credits taken and credits passed data by semester, we computed the difference of total credits taken and passed. The raw data also had information on clubs and other community activities that the student participated in. From this we created a binary variable indicating whether the student participated in community activities or not. For the transfer students we used additional features more relevant to them. Since the mean age of the transfer students (28 years) was much larger we used marital status and highest degree earned as features in the transfer student models.

4.3 Analysis

Student retention data sets are typically imbalanced as the number of dropouts is usually much less than the number that don't. Our dataset was imbalanced with around 35% of dropouts. If the data is imbalanced the standard classifiers have a bias towards the larger majority class. One approach to correcting this imbalance is to preprocess the data in order to balance it out and then build the model. This approach uses various techniques to either oversample the minority class or undersample the majority class or a combination of both. Synthetic Minority Oversampling Technique (SMOTE) is a popular and robust technique that uses a combination of oversampling the minority class and undersampling the majority class which results in better classifier performance [6]. We tried various techniques to correct the imbalance and found the SMOTE technique to yield the best results. Hence our study used SMOTE to correct the imbalance.

We used the features described above and imbalance corrected data to build the following machine learning models: Naïve Bayes, Logistic Regression, Decision Tree, Support Vector Machine, Random Forest, Extreme Gradient Boosting (XGBoost), and a Voting Ensemble model. We chose these models as they are popular classification models some of which have been used in the extant literature. The ensemble model XGBoost has been shown to have high performance in numerous datasets in different domains. The Voting Ensemble is a stacked ensemble model consisting of all of the other models mentioned above. All of the above models were built using 75% of the data for training and 25% for testing and using a 10 fold cross validation to avoid overfitting.

5. RESULTS

The prediction results of the models we trained above are shown in Table 3. Accuracy is a good and intuitive performance measure for balanced datasets. Since we have corrected the imbalance in our dataset prior to building our models, we have reported Accuracy as our measure of performance for the models. The AUC measure and the F1 score were very close to the accuracy measure, so, we have not reported them here, but the information is available on request.

Table 3: Dropout Prediction Results

Model	Accuracy	
	Native Student	Transfer Student
Naïve Bayes	71.3%	40.2%

Support Vector Machine	79.9%	81.2%
Logistic Regression	78.8%	81.4%
Random Forest	79.0%	97.0%
Extreme Gradient Boosting	81.3%	97.1%
Voting Ensemble	80.9%	97.4%

Table 4 shows the features ranked by importance. We have only reported the features that had a score of greater than 1% as the rest have negligible predictive power.

Table 4: Features Ordered by Importance

Native Students		Transfer Students	
Feature	Score	Feature	Score
Cumulative GPA	0.135	SAT English	0.261
High School GPA	0.103	SAT Math	0.191
SAT English	0.100	Cumulative GPA	0.149
SAT Math	0.097	Age	0.093
Difference credits taken and earned	0.095	Difference credits taken and earned	0.063
Age	0.074	Difference credits accepted and applied for transfer	0.047
Gateway English Status	0.051	GPA trend	0.043
English placement	0.046	Community involvement	0.031
Community involvement	0.046	Declared major	0.025
GPA trend	0.046	Developmental Math	0.021
Developmental Math	0.031	GPA trend	0.021
Gender	0.021	Gender	0.013
Race	0.013	Highest degree	0.010

6. DISCUSSION

This study demonstrates how standard machine learning models can be used to predict, with a high degree of accuracy, students at-risk of dropping out. To the best of our knowledge this is one of the first studies to focus on predicting minority student dropout and in particular minority transfer and native student dropout using standard machine learning techniques. The high prediction accuracies achieved in this study demonstrates the effectiveness of standard machine learning models for predicting undergraduate minority student dropout. Reliable identification of at-risk students can help in providing timely interventions to support the student's success and thus increase student retention. Any educational institution can adopt the approach demonstrated in this study with relative ease. A discussion of the various classifiers and the feature importance follows.

The Naïve Bayes classifier, as expected, performed the worst on both the native student and transfer student dataset. In the native student dataset the Extreme Gradient Boosting (XGBoost) model performed the best reaching an 81% accuracy. In the case of transfer students again the XGBoost, Random Forest and the Stacked Voting Ensemble all reached a very high 97% accuracy rate. This very high accuracy rate was indeed surprising as we did not expect to be able to achieve such high performance. All of these models performed much worse without imbalance correction, thus, pointing to the importance of correcting for imbalance in this domain. The results also show that different models perform better between native and transfer students with different features being

important, stressing the importance of modeling these student bodies separately.

For native students their college GPA and high school GPA seemed to be the strongest predictors of dropout. This is consistent with prior literature and the conceptual framework we based the study on. The other factors that had reasonably good predictive power were SAT scores and the difference between credits taken and earned. For transfer students the SAT scores were the strongest predictors with college GPA being the next strongest. Surprisingly High school GPA did not seem to have any impact on transfer student dropout prediction, unlike in the case of native students. This is inconsistent with the literature that has found high school grades to be good predictors of college success [5,15]. One potential explanation could be that the transfer students in our sample are older and thus far removed from high school and hence the high school GPA does not have much predictive power.

Another surprising result was that race did not have much predictive power for both the native students and transfer students. Given the racial gap in graduation rates, even among Black, Hispanic and Asian students, we expected to find some relationship between race and dropout, but found none. Similarly gender did not have much predictive power either. Community involvement was a fairly strong predictor of dropout for both the native and transfer students. This is consistent with the theoretical models of student retention used in our conceptual framework that highlight student engagement as a key factor in college success.

There are several limitations to our current study. The performance results that we have obtained are specific to our sample and caution should be exercised in generalizing them. However, our results do give readers and researchers an idea of the possible accuracy that can be achieved by using standard machine learning models to predict minority student dropout and what features are important. Also, our results can be very informative for institutions with a similar minority student profile as ours. Another limitation is that we have not considered any financial aid data, and other factors not related to academic achievement such as personality traits etc. These are all avenues for further research that we are currently pursuing.

7. CONCLUSION

The results of our study demonstrate that standard machine learning techniques can be effective in predicting minority student dropout with a high degree of accuracy. We also show that different models perform better between transfer students and native students thus reinforcing the importance of modeling these two student bodies separately.

The practical implication of this study is that it demonstrates that educational institutions can use their existing databases that contain routine student data and standard off-the-shelf machine learning models to accurately predict at-risk minority students. This is a cost effective way for institutions to identify at-risk students so that they can devote their resources to offering interventions aimed at retaining them.

8. REFERENCES

- [1] Anglin, L. W., Davis, J. W., & Mooradian, P. W. (1995). Do transfer students graduate? A comparative study of transfer students and native university students. *Community College Journal of Research and Practice*, 19(4), 321-330.
- [2] Bean, J. P. (1980). Dropouts and turnover: The synthesis and test of a causal model of student attrition. *Research in higher education*, 12(2), 155-187.
- [3] Bean, J. P. (1982). Student attrition, intentions, and confidence: Interaction effects in a path model. *Research in Higher Education*, 17(4), 291-320.
- [4] Caison, A. L. (2007). Analysis of institutionally specific retention research: A comparison between survey and institutional database methods. *Research in Higher Education*, 48(4), 435-451.
- [5] Camara, W. J., & Echternacht, G. (2000). The SAT I and high school grades: Utility in predicting success in college. *The College Board Research Notes*, RN-10, 1-12.
- [6] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321-357.
- [7] Davies, T. G., & Casey, K. (1999). Transfer student experiences: Comparing their academic and social lives at the community college and university. *College Student Journal*, 33, 60-71.
- [8] Delen, D. (2010). A comparative analysis of machine learning techniques for student retention management. *Decision Support Systems*, 49(4), 498-506.
- [9] Dougherty, K. (1994). The contradictory college: The conflicting origins, impacts and futures of the community college. Albany, NY: State University of New York.
- [10] Hoffman, J. L. (2002). The impact of student cocurricular involvement on student success: Racial and religious differences. *Journal of College Student Development*, 43 (5), 712-739.
- [11] Kinnick, M. K., & Kempner, K. (1988). Beyond "front door" access: attaining the bachelor's degree. *Research in Higher Education*, 29(4), 299-318.
- [12] Lauría, E. J., Baron, J. D., Deviredy, M., Sundararaju, V., & Jayaprakash, S. M. (2012, April). Mining academic data to improve college student retention: An open source perspective. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (pp. 139-142). ACM.
- [13] Moffat, G. K. (1993, February). The validity of the SAT as a predictor of grade point average for nontraditional college students. Paper presented at the annual meeting of the Eastern Educational Research Association, Clearwater Beach, FL. (ERIC Document Reproduction Service No. ED 356 252)
- [14] Shapiro, D., Dundar, A., Huie, F., Wakhungu, P., Yuan, X., Nathan, A & Hwang, Y., A. (2017, April). Completing College: A National View of Student Attainment Rates by Race and Ethnicity – Fall 2010 Cohort (Signature Report No. 12b). Herndon, VA: National Student Clearinghouse Research Center.
- [15] Schwitzer, A. M., Griffin, O. T., Ancis, J. R., & Thomas, C. R. (1999). Social adjustment experiences of African American college students. *Journal of Counseling and Development*, 77, 189-197.
- [16] Thammasiri, D., Delen, D., Meesad, P., & Kasap, N. (2014). A critical assessment of imbalanced class distribution problem: The case of predicting freshmen student attrition. *Expert Systems with Applications*, 41(2), 321-330.
- [17] Tinto, V. (1993). Building community. *Liberal Education*, 79(4), 16-21.
- [18] Wetzel, J. N., O'Toole, D., & Peterson, S. (1999). Factors affecting student retention probabilities: A case study. *Journal of Economics and Finance*, 23(1), 45-55.

Binary Q-matrix Learning with dAFM

Nan Jiang
Nankai University
Tianjin, China
im.nan.jiang@gmail.com

Zachary A. Pardos
University of California, Berkeley
Berkeley, CA, USA
pardos@berkeley.edu

ABSTRACT

dAFM introduced a neural network implementation of the Additive Factors Model, allowing backpropagation to be used to learn or improve an expert Knowledge Component model. This neural approach learned continuous weights corresponding to item to KC associations. In this work, we extend dAFM to support learning of binary item to KC associations by employing a modification of backpropagation from the literature, called BinaryConnect.

1. INTRODUCTION

An association of items to knowledge components (KC), known as a Q-matrix, is a necessary prerequisite for many models of cognitive estimation used in the educational data mining community and in practice in intelligent tutoring systems [3]. Additive Factors Models (AFM) have been used to refine a Q-matrix by searching through the space of operations (e.g., merge and split) using an expert KC model and other factors of items [4]. Recently, a neural networks approach to modeling and fitting the Q-matrix was introduced which uses backpropagation and stochastic gradient descent to refine the Q-matrix as represented by as a weight coefficient matrix between the item input layer and a hidden layer in the network. This approach, called dAFM [5], results in continuous valued associations between items and KCs. While continuous values in a Q-matrix may open up interesting new interpretations of partial associations, this fuzzy association is not standard. To produce standard binary associations, the weights could be rounded to 1 or 0. This was done in the original dAFM paper but always resulted in accuracy worse than the continuous valued Q-matrix, and often times performed worse than using the original expert Q-matrix. In this paper, we implement a binary weight learning procedure into the fitting process and demonstrate that this approach achieves accuracy better than the original Q-matrix and rivaling that of the continuous valued Q-matrix.

2. RELATED WORK

Several variants on binary weight learning in neural networks have been introduced in the machine learning literature. The primary motivation of these approaches has been to reduce memory size and speed up neural network training, with embedded systems applications in mind. BinaryConnect [1] is one approach that stochastically samples binary weights based on real-values in the forward and back propagation and then updates the weights. Binarized Neural Networks [2] constrain not only weights but also activations to between 1 and -1, which reduces the model size and replaces multiplication with a logical operation. The XNOR-Net [6] approach introduced an extra scaling layer to approximate the parameter and activation by a binary tensor and a scaling factor. Shayer et al.[7] introduced a probabilistic training method, the LR-nets (Local reparameterization networks), which trains the binary weights by introducing a good smoothing approximation and then using its derivative.

As BinaryConnect is an established method and our objective was to get binarized weights in a single specific layer instead of binarizing all weights and activations, we choose BinaryConnect for its simplicity, abundance of public code, and good fit to our task of binarizing Q-matrix learning within dAFM.

3. BINARY CONNECT

BinaryConnect proposes two ways to binarize weights. One is a deterministic sign function, which assigns 1 if the weight is bigger than 0 otherwise assigns -1.

$$w_b = \begin{cases} +1 & \text{if } w > 0, \\ -1 & \text{if otherwise.} \end{cases} \quad (1)$$

The other one is to stochastically assign +1 with probability p and -1 with probability $1-p$ with a 'hard sigmoid' function.

$$w_b = \begin{cases} +1 & \text{with probability } p = \sigma(w), \\ -1 & \text{with probability } 1 - p. \end{cases} \quad (2)$$

$$\sigma(x) = \text{clip}\left(\frac{x+1}{2}, 0, 1\right) = \max\left(0, \min\left(1, \frac{x+1}{2}\right)\right) \quad (3)$$

During the SGD training of binary connect, weights are only binarized in the forward pass and gradient computation, and the float versions of the weights are updated. This is because the real-valued weights are needed in the updating process

Nan Jiang and Zachary Pardos "Binary Q-matrix Learning with dAFM" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 588 - 590

in SGD.

4. BINARY dAFM

Our goal is for the associations between questions and knowledge components (KC) to be represented by the more interpretable binary instead of continuous weights; accordingly, we implement BinaryConnect into dAFM and describe the technical implementation in this section. Considering the real meaning of the associations, we constrain the weights to 1 and 0 instead of 1 and -1. The binary weight learning adapted to dAFM is as follows:

Algorithm SGD training with BinaryConnect in dAFM. C is the cost function and `binarize()` and `clip()` are the functions that binarize and clip weights.

Require: previous weights in the QK layer WB_{qk} , Qk layer, the layer before the QK layer—the step_input layer, b_{t-1} : bias and learning rate η .

Ensure: updated parameter w_t and b_t

1. Forward propagation:

$WB_{qk} \leftarrow \text{binarize}(Wqk_{t-1})$
compute QK knowing step_input, WB_{qk} , b

2. Backward propagation:

get the Qjk layer's activations gradient $\frac{\partial C}{\partial(qk)}$
compute $\frac{\partial C}{\partial(\text{step_input})}$ knowing $\frac{\partial C}{\partial(qk)}$ and WB_{qk}

3. Parameter update:

compute $\frac{\partial C}{\partial(WB_{qk})}$ and $\frac{\partial C}{\partial(Qk)}$ and step_input
 $Wqk_t \leftarrow \text{clip}(Wqk_{t-1} - \eta \frac{\partial C}{\partial(WB_{qk_{t-1}})})$
 $b_t \leftarrow b_{t-1} - \eta \frac{\partial C}{\partial(b_{t-1})}$

Programatically, this translated to the following modifications to the dAFM code, including definition of a BinaryDense neural network layer and Binarization function:

```
if binary=="False":
    Q_jk = TimeDistributed(Dense(skills,
        activation="linear",
        kernel_initializer=self.f(Q_jk_initialize),
        use_bias=False, trainable=qtrainable),
        trainable=qtrainable, name="Q_jk")(step_input)
else:
    Q_jk = TimeDistributed(BinaryDense(skills,
        activation="linear",
        kernel_initializer=self.f(Q_jk_initialize),
        use_bias=False, trainable=qtrainable),
        trainable=qtrainable, name="Q_jk")(step_input)
```

BinaryDense layer

```
self.w = self.add_weight(shape=(input_dim, self.units),
    initializer=self.w_initializer,
    name="w",
    regularizer=self.w_regularizer,
    constraint=self.w_constraint)
self.binary=binarize(self.w, H=self.H)
output = K.dot(inputs, self.binary)
```

Binarize function

```
def binarize(W, H=1):
    Wb = H * binary_tanh(W / H)
    return Wb
def round_through(x):
    rounded = K.round(x)
    return x + K.stop_gradient(rounded - x)
def _hard_sigmoid(x):
```

```
x = (0.5 * x) + 0.5
return K.clip(x, 0, 1)
def binary_tanh(x):
    return round_through(_hard_sigmoid(x))
```

In our binary dAFM Model implementation, we integrate a Keras implementation¹ of a BinaryDense layer. Specifically, we manage the backpropagation of the binarized weights and update the real-valued weights using the `tf.stop_gradient()` function.

5. RESULTS

Results of dAFM models' predictions using a 30% test set hold-out on our four primary datasets are shown in Table 1. The AFM model represents the baseline standard model with its expert Q-matrix. The Binary dAFM model (hard sigmoid) represents the model that uses the hard sigmoid function and Binary dAFM (sign function) represents the model that use the sign function to binarize. The RMSE is reported on the model's prediction of the correctness of student first attempt responses to items in datasets from the Cognitive Tutor and ASSISTments.

We can observe that the original dAFM model with continuous valued Q-matrix associations beats the original expert KC model (AFM) in all datasets, a result observed in the original paper. We can also observe that the hard sigmoid variants of Binary dAFM performed better than the sign function variant in three of the four datasets. Additionally, the hard sigmoid Binary dAFM performed better than the original AFM model in all datasets, a departure from the rounded approach which performed worse than AFM on the majority of datasets. In one case, with ASSISTments 2012-2013, the Binary approach even beats continuous valued dAFM, scoring a substantially lower RMSE of 0.4339 compared to 0.4443. We conclude from these observations that the BinaryConnect approach to binarized weight learning is an effective alternative approach to continuous Q-matrix association representation with neural networks.

5.1 Replication

1. Clone the github repository.

```
1 git clone https://github.com/CAHLR/dAFM.git
```

2. Enter into dAFM directory using following command.

```
1 cd dAFM
```

3. Execute the `src/main.py` script using python after specifying the input values. It will train on a specified portion of the dataset serving as the training set and evaluate the average root mean square error on a specified validation set portion. By Passing different values as arguments, different datasets and models can be trained and evaluated.

```
1 python3 src/main.py --dataset G0 --dataset_path
2 G0/Geometry96-97 --user_id 'Anon Student Id' --
3 problem_id 'question' --correctness 'test' --sk
4 ill_name 'KC (Original)' --dAFM fine-tuned No --
5 save_model False --binary True
```

See the github README for additional options

¹https://github.com/DingKe/nn_playground

Table 1: Results of the Expert KC model (AFM), continuous dAFM (dAFM), and Binary dAFM models (hard and sigmoid and sign function variants). Response prediction error metric is RMSE.

Dateset	AFM	dAFM	Binary dAFM(hard sigmoid)	Binary dAFM(sign function)
Geometry	0.4355	0.4016	0.4263	0.4303
ASSISTments 09-10	0.4777	0.4594	0.4608	0.4596
ASSISTments 12-13	0.4504	0.4443	0.4339	0.4367
CogTutor Bridge 06-07	0.3698	0.3655	0.3656	0.3702

6. REFERENCES

- [1] M. Courbariaux, Y. Bengio, and J.-P. David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*, pages 3123–3131, 2015.
- [2] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In *Advances in neural information processing systems*, pages 4107–4115, 2016.
- [3] K. R. Koedinger, S. D’Mello, E. A. McLaughlin, Z. A. Pardos, and C. P. Rose. Data mining and education. *Wiley Interdisciplinary Reviews: Cognitive Science*, 6(4):333–353, 2015.
- [4] K. R. Koedinger, E. A. McLaughlin, and J. C. Stamper. Automated student model improvement. In K. Yacef and O. Zaiane, editors, *Proceedings of the 5th International Conference on Educational Data Mining*, pages 17–24, 2012.
- [5] Z. A. Pardos and A. Dadu. dafm: Fusing psychometric and connectionist modeling for q-matrix refinement. *Journal of Educational Data Mining*, 10(2):1–27, 2018.
- [6] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016.
- [7] O. Shayer, D. Levi, and E. Fetaya. Learning discrete weights using the local reparameterization trick. *arXiv preprint arXiv:1710.07739*, 2017.

A Comparative Analysis of Emotional Words for Learning Effectiveness in Online Education

Jaechoon Jo
Sangmyung University
Dept. of Smart Information
Communication Engineering
Cheonan, South Korea
(+82)41-550-5217
jae@smu.ac.kr

Yeongwook Yang
Korea University
Dept. of Computer Science and
Engineering
Seoul, South Korea
(+82)2-3290-2684
yeongwook@blp.korea.ac.kr

Gyeongmin Kim
Korea University
Dept. of Computer Science and
Engineering
Seoul, South Korea
(+82)2-3290-2684
totoro4007@gmail.com

Heuseok Lim*
Korea University
Dept. of Computer Science and
Engineering
Seoul, South Korea
(+82)2-3290-2396
limhseok@korea.ac.kr

ABSTRACT

To overcome space and time limits in education environment, online education is becoming increasingly crucial. Most online education is based on video lectures. Video-based lectures are delivered to the learner through the language of the instructor. Hence, the language (word) of the instructor affects learners during the video discourse. In this study, we examined the effectiveness between emotional (positive and negative) and non-emotional (neutral) words for learning effectiveness on online education based on video lectures. The results indicated that learners remember emotional words better than non-emotional ones. If educational content based on video lectures is implemented using emotional words, then it will improve the learning effectiveness in online education through video lectures.

Keywords

Video Lecture, Online Education, Gamification, Short-Term Memory

1. INTRODUCTION

The scale of online education service market is gradually expanding at present because of the development of technology and interest in education. The demand for online education is also increasing as education transcends regional and temporal boundaries. An example of these phenomena is massive open online course (MOOC). MOOC refers to online learning that

allows anyone to enroll in free courses on a large scale without a restriction in the number of students. In 2008, MOOC offered educational opportunities to learners and allowed them to take lectures from prestigious universities online [1].

Online education services, such as MOOC, facilitate learning through video lectures; thus, learners acquire knowledge by watching the video lectures [2,3]. In such an environment, determining whether the learner has actually watched the video lecture or not is important. This problem is a considerably challenging task because we can monitor the learner based only on a limited range of data, that is, the interaction between the learner and computer.

The common technique in addressing this issue is to assess whether the learner is watching the video lecture by solving the relevant quizzes or simply checking the playing time of the video. The method of judging whether a learner watches a video lecture through quizzes can evaluate the learner's understanding level. However, constructing relevant quizzes and completing these quizzes are time consuming for both the instructor and learners, respectively [4]. Moreover, the sharing of quizzes among learners may lose its original purpose. Hence, the method of checking the playing time of the video is not an accurate evaluation technique because it recognizes that the learner has watched the video even when the learner is performing other tasks or is absent. Through the short-term memory, we can assess whether learners watch the video lecture or not. Therefore, in this study, we developed a short-term memory judge system.

This short-term memory judge system determines whether or not a learner has watched the video lecture through cognitive processing based on the frequency effect of the word used in the video lecture. In this work, we consider that emotional words can improve a learner's memory better than non-emotional ones; hence, we analyzed the effectiveness of emotional words. The

Jaechoon Jo, Yeongwook Yang, Gyeongmin Kim and Heuseok Lim "A Comparative Analysis of Emotional Words for Learning Effectiveness in Online Education" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 591 - 594

results indicated that even in the short-term memory judge system, learners remember emotional words better than neutral ones.

The paper is organized as follows: Section 2 presents research related to this study; Section 3 describes the short-term memory judge system; Section 4 discusses experimental evaluation; and, finally, Section 5 provides a conclusion as well as notes on future research.

2. RELATED WORK

2.1 Learning Judge System

Online education services are based on video lectures, and various studies have been conducted to enhance learning experience in self-directed learning environments. Jang (2011) analyzed the learning attitudes of e-learning students through electroencephalogram (EEG) and provided feedbacks according to the learner's learning attitude. He also examined the correlation between the learning attitude and EEG to improve the learning efficiency of students. Consequently, distinguishing when to concentrate on learning and when not is possible. Another limitation is that actual EEG equipment is necessary and is influenced by the surrounding environment [5]. Lee (2009) developed a platform to gauge the response speed to voice and visual stimuli, assessed the language performance effectively, and applied it to improve learning. The platform concurrently measured the speed and accuracy for language processing. Hence, speed and accuracy were observed to enhance language learning [6]. Kim (2014) analyzed the patterns of thousands of learners watching online video lectures and determined the dropout and interaction peak. Accordingly, the high dropout rate was noticed in videos with long playback time. Hence, video playback time can be an important factor affecting the learning outcome [7]. Mills (2011) analyzed how mind-wandering occurs during learning using an intelligent tutoring system. The result of the experiment indicated that an average of 11.5 mind-wandering occurred with an interval of 2 min [8]. Given that mind-wandering frequently occurs in an online education environment, automatically detecting the occurrence of this event is also crucial. Various studies and methods have been conducted and proposed to enhance learning effectiveness in an online education environment. However, they remain remarkably challenging because the learning effectiveness can be assessed based only on very limited data, that is, the interaction data between the learners and computers only.

2.2 Emotional Words

Research on the memory between emotional and neutral words has been conducted in the field of cognitive psychology. Kensinger & Corkin (2003) and Ochsner (2000) reported that emotional words are easier to be recalled than neutral words [9,10]. Another research was conducted in the field of natural language processing to classify whether sentences have a positive or negative meaning through emotional words [11]. Esuli and Andrea et al. (2007) proposed *SentiwordNet* to efficiently categorize the positive and negative meaning of words [12]. In this work, we evaluated the difference of memory retention between emotional and neutral words using the short-term memory judge system. We also utilized the existing *Korean Emotional Word Dictionary* to distinguish the difference among positive, negative, and neutral words [13].

3. SHORT-TERM MEMORY JUDGE SYSTEM

The short-term memory judge system can automatically detect whether mind-wandering occurs in video lecture-based online education. To detect the short-term memory, the system determines whether mind-wandering occurs while the learner is watching the video lecture. To assess whether mind-wandering occurs, we applied the word frequency effect theory of cognitive psychology. The word frequency effect indicates the difference in cognitive responses between high- and low-frequency words. The theory is that a difference in cognitive responses exists between the words that people actually know and the words they do not recognize. The system judges the short-term memory by calculating the learner's word recognition response through a word game. The word game spontaneously generates high-frequency words of the video lecture viewed by the learner and words not set in the video lecture. Moreover, the learner can actually see a randomly displayed word from the automatically generated word set in the word game and selects the correct answer. Figure 1 shows developed word game UI.

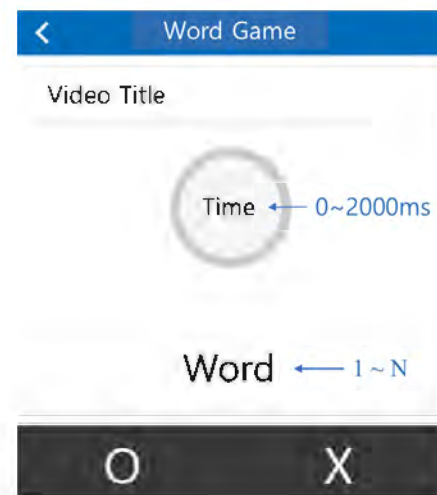


Figure 1. Word Game UI

The short-term memory judge system is automatically determined as Pass or Fail based on the correct word rate and response rate in the word game. According to cognitive psychology, the speed of recognizing words that people know is between 0.7 and 1.2 s. Therefore, the existing cognitive response rate theory was applied as a criterion for the detection. Considering that the word game is implemented in web environment, 0.3–1.7 s is set as the word recognition response criterion, which was added with 0.4–0.5 s [14].

4. EXPERIMENTAL EVALUATION

4.1 Dataset

To verify the idea of this study, data were collected using short-term memory judge system from 576 students who take the university major courses in Korea. The data contain user information, video lecture information, frequent words, response time, and correctness information. By using the response time and correctness rate, we determined whether or not the learner has actually watched the video lecture. The number of video lectures

used in the system is 77, and the number of words used is 17,592 (Table 1). We classified the emotional words into positive and negative and classified non-emotional words into neutral. These data have a total of 3,158 positive, 3,283 negative, and 11,151 neutral words [13]. Generally, considering that neutral words are more common than emotional ones, the sum of the neutral words is much higher than the emotional words in this data set. The list of positive, negative, and neutral words is presented in Table 2. The total number of the gathered data is 47,517.

Table 1. Data sets of the positive, negative, and neutral words

# of data sets	# of users	# of contents	# of words
47,517	576	77	17,592

Table 2. Example of positive, negative, and neutral words

Positive	Negative	Neutral
Happy	Destroy	Office
Thanks	Fight	Research
True	Difficult	Drama
Joy	Lie	Bank
Positive	Unhappy	Apartment

4.2 Analysis & Results

We have calculated the average response time and correctness of each emotional word to analyze the effectiveness of emotional words using the short-term memory judge system. High correctness value indicates that the learner memorizes the word well from the lecture, whereas the low response time denotes that the learner rapidly responded to the word. The calculated results are presented in Table 3.

Table 3. Average of correctness and response time per emotional word

	Correctness (%)	Response Time (ms)
Positive	0.811	919.7
Negative	0.688	924.2
Neutral	0.519	898.3

The interpretation of the calculated results should consider both the correctness and time. Correctness is the highest with a positive word of 0.811. The negative word has lower correctness than the positive word but is better than the neutral word. Meanwhile, the neutral word has the lowest score of 0.519. Response time is the fastest with the neutral word at 898.3ms and slowest with the negative word at 924.2ms. However, it can be interpreted that around 900ms are word recognition time according to cognitive psychology.

The analyzed result indicates that emotional words are better remembered by learners than non-emotional words. Alternatively, constructing video-based education content with emotional words can enhance learning effectiveness because it reflects the learner's memory better than that with non-emotional words.

5. CONCLUSION

To examine the effectiveness of emotional information of words by using the short-term memory judge system, we have analyzed the data based on emotional information of words applying the data gathered from the system. Finally, we can confirm that emotional (positive and negative) words have more influence on learners than non-emotional (neutral) words. When we construct an educational content based on video lecture through emotional words, we conclude that this method can promote considerably effective learning.

Our future work includes constructing a system that introduces word emotional information to the short-term memory judge system and analyzes the effectiveness of word emotional information profoundly through experimentation.

6. ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government [grant number NRF-2018R1D1A1B07051369]

7. REFERENCES

- [1] Joo, Y. and D. Kim. 2017. A Study of Satisfaction and Intention to Use MOOC Based on UTAUT2 in Korea. *Journal of Lifelong Learning Society*. 13, 1, 185-207.
- [2] Fischer, G. 2014. Beyond hype and underestimation: identifying research challenges for the future of MOOCs. *Distance education*. 35, 2, 149-158.
- [3] Hollands, F. M. and D. Tirthali. 2014. Why do institutions offer MOOCs?. *Journal of Asynchronous Learning Networks*. 18, 3, 1-19.
- [4] Jo, J., Yu, W., Koh, K. H., and Lim, H. 2017. Development of a Game-Based Learning Judgment System for Online Education Environments Based on Video Lecture: Minimum Learning Judgment System. *Journal of Educational Computing Research*. 56, 6, 802-825.
- [5] Jang, J. K. and H. S. Kim. 2011. EEG Analysis of Learning Attitude Change of Female College Student on e-Learning. *Journal of the Korea Contents Associations*. 11, 4, 42-50.
- [6] Lee, H. and S. Beak. 2010. Reaction Test Platform and Application by Auditory and Visual Stimulus for Language Learning Ability Improvement. *Journal of Korean Society for Internet Information*. 11, 1, 77-84.
- [7] Kim, J., P. J. Guo, D. T. Seaton, P. Mitros, K. Z. Gajos, and R. C. Miller. 2014. Understanding in-video dropouts and interaction peaks in online lecture videos. *In Proceedings of the first ACM conference on Learning @ scale conference (L@S '14)*. ACM, New York, NY, USA, 31-40.
- [8] Mills, C., D'Mello, S., Bosch, N., and Olney, A. M. 2015. Mind wandering during learning with an intelligent tutoring system. *In International Conference on Artificial Intelligence in Education*. 267-276.
- [9] Kensinger, E. A., and Corkin, S. 2003. Memory enhancement for emotional words: Are emotional words more vividly remembered than neutral words?. *Memory & Cognition*. 31, 1169-1180.
- [10] Ochsner, K. N. 2000. Are affective events richly recollected or simply familiar? The experience and process of recognizing feelings past. *Journal of Experimental Psychology: General*. 129, 242-261.

- [11] Pang, B, and Lillian L. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*. 2, 1-2, 1-135.
- [12] Esuli, Andrea, and Fabrizio Sebastiani. 2007. SentiWordNet: a high-coverage lexical resource for opinion mining. *Evaluation*. 17, 1-26.
- [13] H. Shin, M. Kim, Y. Jo, H. Jang, and A. Cattle. 2012. Annotation Scheme for Constructing Sentiment Corpus in Korean. *In proceedings of the 26th Pacific Asia Conference on Language, Information and Computation*. 181-190.
- [14] Jo, J., and Lim, H. 2016. How to Judge Learning on Online Learning: Minimum Learning Judgment System. *Educational Data Mining*. 597-598.

Identifying Critical Pedagogical Decisions through Adversarial Deep Reinforcement Learning

Song Ju, Guojing Zhou, Hamoon Azizsoltani, Tiffany Barnes, Min Chi
Department of Computer Science
North Carolina State University
Raleigh, NC 27695
{sju2, gzhou3, hazizso, tmbarnes, mchi}@ncsu.edu

ABSTRACT

For many forms of e-learning environments, the system's behaviors can be viewed as a sequential decision process wherein, at each discrete step, the system is responsible for deciding the next system action when there are multiple ones available. Each of these system decisions affects the user's successive actions and performance and some of them are more important than others. Thus, this raises an open question: how can we identify the critical system interactive decisions that are linked to student learning from a long trajectory of decisions? In this work, we proposed and evaluated Critical-Reinforcement Learning (Critical-RL), an adversarial deep reinforcement learning (ADRL) based framework to identify critical decisions and induce compact yet effective policies. Specifically, it induces a pair of adversarial policies based upon Deep Q-Network (DQN) with opposite goals: one is to improve student learning while the other is to hinder; critical decisions are identified by comparing the two adversarial policies and using their corresponding Q-value differences; finally, a Critical policy is induced by giving optimal action on critical decisions but random yet reasonable decisions on others. We evaluated the effectiveness of Critical policy against a random yet reasonable (Random) policy. While no significant difference was found between the two condition, it is probably because of small sample sizes. Much to our surprise, we found that students often experience so-called *Critical phase*: a consecutive sequence of critical decisions with the same action. Students were further divided into High vs. Low based on the number of Critical phases they experienced and our results showed that while no significant was found between the two Low groups, the High Critical group learned significantly more than the High Random group.

Keywords

Reinforcement Learning, Critical Decision, Deep Q-Network, Adversarial Reinforcement Learning

Song Ju, Guojing Zhou, Hamoon Azizsoltani, Tiffany Barnes and Min Chi "Identify Critical Pedagogical Decisions through Adversarial Deep Reinforcement Learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 595 - 598

1. INTRODUCTION

Intelligent Tutor Systems (ITSs) are a type of highly interactive e-learning environment which facilitates learning by providing contextualized feedback and step-by-step support to individual students [4, 14]. These step-by-step behaviors can be viewed as a sequential decision process where at each step the system chooses an action (e.g. give a hint, show an example) from a set of options. During tutoring, the system makes a series of decisions to provide adaptive instructions. Some of the decisions might be *more important and impactful than others*. This raises a major open question: ***How can we identify the critical system interactive decisions that are linked to student learning especially in a long trajectory of decisions?*** For example, in our ITS, the tutor makes more than 400 sequential decisions during training.

Reinforcement Learning (RL) offers one of the most promising approaches to data-driven decision-making applications and RL algorithms are designed to induce effective policies that determine the best action for an agent to take in any given situation so as to maximize some predefined cumulative reward. A number of researchers have studied the application of existing RL algorithms to improve the effectiveness of ITSs [2, 13, 12, 3]. However, relatively little work has been done to analyze, interpret, and explain RL-induced policies. While traditional hypothesis-driven, cause-and-effect approaches offer clear conceptual and causal insights that can be evaluated and interpreted, RL-induced policies are often large, cumbersome, and difficult to understand. In this work we propose to induce compact RL policies that highlight key or critical decisions by taking advantage of the structure of the domain and the structure of our induced policies by leveraging the conditional independence relationships among the state features.

We propose Critical-RL, an adversarial deep reinforcement learning (ADRL) based framework to induce compact policies that make critical decisions. By inferring critical decisions we can identify which tutor actions are minimally necessary for the tutoring process to be effective, which holds great implications for systems research. Additionally, inference of critical relationships is one of the central tasks of science and it is one of the most challenging topics in many disciplines, particularly in the areas where controlled experiments are comparatively expensive or even impossible. In this work, we propose a general framework that fully integrates automatic critical inference and standard

reinforcement learning in an ITS setting. We expect that our framework can be spread to other similar domains for related critical tasks.

2. RELATED WORK

2.1 Applying RL to ITSs

Prior work has applied a variety of RL approaches to induce pedagogical policies to improve the effectiveness ITS [1, 6, 15, 10]. For example, Beck et al. [1] applied temporal difference learning on simulated students to induce a policy that would minimize student time on task. Results showed that the policy group indeed spent significantly less time than the non-policy group. Shen et al. [13] applied MDP to induce a pedagogical policy aimed at improving students' learning performance. Results showed that the induced policy was significantly more effective than a random baseline policy for certain learners. Mandel et al. [6] applied a POMDP based approach to induce a pedagogical policy targeted at improving students' learning gain. The RL-induced policy was then compared with an expert policy and a random baseline policy. Results revealed that the RL-induced policy significantly outperformed the other two. Wang et al. [15] applied a variety of deep RL approaches on simulated students to induce pedagogical policies that would improve students' normalized learning gain in an educational game. Simulation results suggested that deep RL policies were more effective than linear model based RL policies.

To summarize, prior work has shown that RL induced policies can lead to improved student learning/behavior as compared to baseline policies. However, prior work has mainly focused on inducing effective policies from pre-collected data or simulated student, but put relatively less effort to identify the exact part that makes them effective.

2.2 Deep Reinforcement Learning (DRL)

Recent advance in deep learning has allowed RL to work in complex interactive environments which was often impractical in before. Recent work showed that RL can induce effective policies for a variety of tasks, such as game playing [8, 9], robotic control [5, 19], recommendation generation [17, 16] and also ITS control [15, 10]. However, all of the state-of-art RL algorithms focused on inducing effective policies. None of them considered interpreting, explaining and identifying critical decisions from RL induced policies.

2.3 Exploiting Q-value Difference

Some prior work has exploited the Q-value difference between actions to simplify the decision-making process/problem. For example, Mitchell et al. [7] relied on the Q-value difference to select features for RL. They proposed a Q-value difference based policy evaluation metric, which was then used to guide feature selection for RL. Zhou et al. [18] relied on Q-value difference to reduce the policy space. More specifically, they applied weighted decision tree with post-pruning to extract a compact set of 529 rules from a full set of 3706 rules. During the extraction, each rule was weighted by the Q-value difference between two alternative actions and thus increased the carry-out likelihood of more important decisions. Results showed that the full RL policy and the compact DT policy together were significantly more effective than a random policy and there is no significantly

difference between the full RL policy and the compact DT policy.

In sum, prior studies have used Q-value difference to measure action importance and results suggest that it is an effective measure. However, prior work used Q-value difference to reduce the feature space or the policy space, but we used it to reduce the decision space.

3. METHOD

3.1 Adversarial Reinforcement Learning

Adversarial Reinforcement Learning (ARL) is a category of RL which can induce a pair of policies for opposite goals. In our application, an *Original Policy* was induced using the original rewards and an *Inversed Policy* was induced using the inversed rewards, which is the negative value of the original rewards. We expect these two policies to have opposite goals, one to help student learn while the other to hinder them learn.

3.2 Deep Q-Network (DQN)

Deep Q-Network (DQN) is a RL algorithm that uses a deep neural network to approximate the Q-value function. The neural network takes a state as input, which is represented as a numerical vector, and outputs its estimation of the Q values for all possible actions. During training, the neural network is updated recursively following the Bellman equation shown below until converge.

$$Q_{i+1}(s, a) = \mathbb{E}_{s' \sim \epsilon} [r + \gamma \max_{a'} Q_i(s', a') | s, a] \quad (1)$$

where γ is a discount factor, ϵ is the environment and Q_i is the action-value function at the i th iteration. DQN is a *model free* approach that it is focused on estimating the action value functions from the interactions with the environment without constructing a model of the environment. Also, it is an off-policy approach that the new policy is induced based upon the historical data generated by an alternative behavior policy.

3.3 Identifying Critical Decision

Once the adversarial policies are induced, critical decisions are identified following two rules: 1) given the state, the two policies make opposite decisions and 2) the decision is important for both policies.

For a given state, rule one is tested first. If the two policies make the same decisions, it is not critical. Otherwise, rule two is tested. In order to measure the importance of the decision for each policy, we calculate the absolute Q-value difference between the two alternative actions: $\Delta Q^*(s) = |Q^*(s, a_1) - Q^*(s, a_2)|$. If this difference is greater than a threshold, the decision is considered important for the corresponding policy. In this paper, we set the threshold to be the median Q-value difference for all decisions in our training data set.

3.4 Critical Policy Induction

In tutoring, our ITS provides students with the same 12 problems in the same order. Among them, the first and the

eight problems are fixed to be problem solving where the students is required to solve all the steps. For the rest 10 problems, the policies decide whether to elicit the next step from the student or to directly show the student how to solve the next step.

Thus, we induced 10 pairs of adversarial policies, one for each problem. Each pair of the adversarial policies consist of two policies: an original policy and an inversed policy. The original policy was induced using the original rewards while the inversed policy was induced using inversed rewards. Other than the rewards, all other parts of the data were identical, such as state representation and transition samples.

In order to find the best policy, for each problem, we implemented two different types of neural network: Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) to induce the adversarial policies. The policies were then evaluated using Per decision importance sampling (PDIS)[11] and the better one was selected. Once the adversarial policies were induced, whether a decision was critical or not during tutoring was determined following the two rules mentioned in section 3.3.

Finally, the Critical policy is carried out partially in that if a decision is critical, it will be carried out; otherwise, the decision will be made randomly. More specifically, for a given state, the adversarial policies are queried to determine whether the decision is critical. If it is, the decision made by the original policy will be taken; otherwise, a random decision will be taken.

4. EXPERIMENT SETUP

In order to evaluate the critical-RL induced policy, we conducted a classroom study comparing the Critical policy with the Random policy. The participants of this study were undergraduate students enrolled in the Discrete Mathematics class at the Department of Computer Science at NC State University in 2018 Fall. In this study, all students were required to complete 4 phases: 1) pre-training, 2) pre-test, 3) training on Pyrenees tutor, and 4) post-test. Pyrenees tutor is a web-based ITS for probability, which covers 10 major principles of probability such as the Complement Theorem and Bayes' Rule. During the experiment, all students in both two conditions studied the same materials, received the same questions in pre-test, trained on the same tutor, examined the same questions in post-test. The only difference was the policies used in the tutor.

In this study, 120 students were randomly assigned to the Critical condition and the Random condition. Due to preparation for final exams and the length of the study, 96 student completed the study. 3 students performed perfectly in the pre-test were excluded from our subsequent statistical analysis. The final group sizes were: $N = 50$ (Critical) and $N = 43$ (Random). We performed a Chi-square test of the relationship between students' condition and their completion rate and found no significant difference between the conditions: $\chi^2(1) = 2.55, p = 0.11$.

5. RESULTS

Table 1 shows the mean and standard deviation (SD) of the post-test score, learning gain (LG) and total training time for the Crucial and Random condition. Contrast comparison analysis showed no significant difference between the two conditions on all three measures. Please note that although the Critical condition appeared to outperform the Random condition on learning gain (0.05 vs. 0.03), such difference was not significant ($p = 0.50$). One of the possible reasons is that the group size was not large enough to demonstrate significance. A post hoc power analysis revealed that a total sample of 1544 students was required to detect significance at .05 on small effects ($d=.14$), with 80% power using a contrast.

Table 1: Critical vs. Random

Measure	Critical	Random	P value
Post	0.71 (0.19)	0.71 (0.20)	0.91
LG	0.05 (0.18)	0.03 (0.14)	0.50
Time	121.6 (37.7)	116.3 (30.47)	0.46

Since the Critical policy was partially carry-out where only critical decisions were made following the optimal policy, we conducted an inspection into the relation between the number of critical decisions made and student learning. Through analyzing the student-system interactive logs, we found that critical decisions were always appeared in groups and each group of consecutive critical decisions had the same action. This is aligned with existing learning theory that the learning process is a continuous process. Student can stay in the same learning state during several steps but this continuous learning state is hard to be represented by current features. In other words, in the same learning state, the agent will continuous giving same actions to the student until he moves to next learning state. So, we defined *Critical Phase* as a period of consecutive critical decisions executions with the same action according to the Critical policy.

In order to analyze the impact of critical phase, we divided students into High vs. Low groups by a median split on the number of critical phases they experienced. For the Random condition, as the execution of decisions were partially agreed with the Critical policy, we ignored the actual decision and only focused on the Critical policy's decision. Thus, we had four groups based upon their critical phase number and policies: High-Random ($n=20$), Low-Random ($n=23$), High-Critical ($n=27$), Low-Critical ($n=23$). A two-way ANOVA analysis using policies {Critical, Random} and critical phase {High, Low} as two factors and the student's learning gain as the dependent measure showed a significant interaction effect $F(1, 89) = 7.163, p = 0.009$. Subsequent contrast analysis revealed that the High-Crucial group ($M = 0.098, SD = 0.2$) significantly outperformed High-Random group ($M = -0.011, SD = 0.16$): $t(89) = 2.360, p = 0.02$. However, such difference was not significant between the Low-Crucial group ($M = 0.001, SD = 0.13$) and Low-Random group ($M = 0.067, SD = 0.12$): $t(89) = 1.42, p = 0.16$.

In terms of time on task, a two-way ANOVA analysis on condition and critical phase number showed a main effect on critical phase number: $F(1, 89) = 5.579, p = 0.020$ in

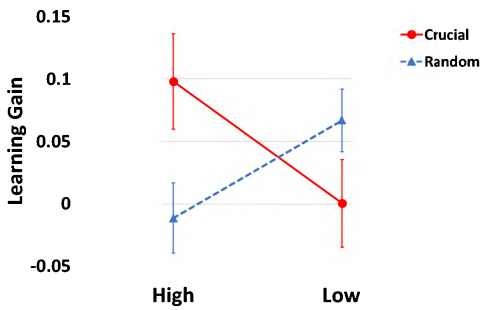


Figure 1: Comparison of Learning Performance

that the High group ($M = 127.51, SD = 36.34$) spent significantly more time on task than the Low group ($M = 110.56, SD = 30.51$). The results suggest that students experienced more critical phases are more likely to spend more time on task.

6. CONCLUSION

In this study, we proposed Critical-RL to identify critical pedagogical decisions in an ITS. Based on the ADRL framework, we induced a Critical policy which gives optimal action on critical decision points but randomly select actions on others. We empirically compared the Critical policy with a baseline Random policy in a classroom study for real students. Although there's no significant difference between the two conditions, we found the existence of Critical phase, a consecutive sequence of critical decisions with the same action. We then divided students into High vs. Low groups based on the number of Critical phases they experienced. Results showed that while the two Low groups were not sensitive to pedagogical policies, the High-Critical group significantly outperformed the High-Random group. This suggested that for certain students, the Critical policy is significantly more effective than the Random policy.

In the future, we plan to analyze the difference between states in critical and non-critical phase. Through analyzing the critical state, we hope to align critical decision with existing learning theory and further generalize our method to other domain.

7. ACKNOWLEDGMENTS

This research was supported by the NSF Grants #1432156, #1651909, #1726550, and #1916417.

8. REFERENCES

- [1] J. Beck, B. P. Woolf, and C. R. Beal. Advisor: A machine learning architecture for intelligent tutor construction. In *AAAI/IAAI*, pages 552–557, 2000.
- [2] M. Chi, K. VanLehn, D. Litman, and P. Jordan. Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *User Modeling and User-Adapted Interaction*, 21(1-2):137–180, 2011.
- [3] B. Clement, P.-Y. Oudeyer, and M. Lopes. A comparison of automatic teaching strategies for heterogeneous student populations. In *EDM*, 2016.
- [4] K. R. Koedinger, J. R. Anderson, W. H. Hadley, and M. A. Mark. Intelligent tutoring goes to school in the big city. 1997.
- [5] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *JMLR*, (17(39)):1–40, 2016.
- [6] T. Mandel, Y.-E. Liu, S. Levine, E. Brunskill, and Z. Popovic. Offline policy evaluation across representations with applications to educational games. In *AAMAS*, pages 1077–1084, 2014.
- [7] C. M. Mitchell, K. E. Boyer, and J. C. Lester. Evaluating state representations for reinforcement learning of turn-taking policies in tutorial dialogue christopher. *SIGDIAL*, pages 339–343, 2013.
- [8] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*, 2013.
- [9] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, and et al. Human-level control through deep reinforcement learning. *Nature*, (518):529–533, 2015.
- [10] K. Narasimhan, T. Kulkarni, and R. Barzilay. Language understanding for text-based games using deep reinforcement learning. *arXiv preprint arXiv:1506.08941*, 2015.
- [11] D. Precup, R. S. Sutton, and S. Singh. Eligibility traces for off-policy policy evaluation. *ICML*, pages 759–766, 2000.
- [12] A. N. Rafferty, E. Brunskill, T. L. Griffiths, and P. Shafto. Faster teaching via pomdp planning. *Cognitive science*, 40(6):1290–1332, 2016.
- [13] S. Shen and M. Chi. Reinforcement learning: the sooner the better, or the later the better? In *the 2016 Conference on User Modeling Adaptation and Personalization*, pages 37–44. ACM, 2016.
- [14] K. Vanlehn. The behavior of tutoring systems. *International journal of artificial intelligence in education*, 16(3):227–265, 2006.
- [15] P. Wang, J. Rowe, W. Min, B. Mott, and J. Lester. Interactive narrative personalization with deep reinforcement learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017.
- [16] X. Zhao, L. Xia, L. Zhang, Z. Ding, D. Yin, and J. Tang. Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems.*, pages 95–103, 2018.
- [17] G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie, and Z. Li. Drn: A deep reinforcement learning framework for news recommendation. *World Wide Web Conference*, pages 167–176, 2018.
- [18] G. Zhou, J. Wang, C. F. Lynch, and M. Chi. Towards closing the loop: Bridging machine-induced pedagogical policies to learning theories. *EDM*, 2017.
- [19] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. *ICRA*, 2017.

Smart Learning Object Recommendations based on Time-Dependent Learning Need Models

Christopher Krauss
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
christopher.krauss
@fokus.fraunhofer.de

Agathe Merceron
Beuth University for Applied
Sciences
Luxemburger Str. 10
13353 Berlin, Germany
merceron
@beuth-hochschule.de

Stefan Arbanowski
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
stefan.arbanowski
@fokus.fraunhofer.de

ABSTRACT

This paper deals with adaptive learning technologies that fit the individual learner's needs. Thereby, recommender systems play a key role in supporting the user's decision process for an efficient and effective item selection. Activity data have been collected from students using course materials available online. The courses provided access to the course materials via a novel web application. This app also presented learning recommendations to make the content selection more efficient and effective. The paper focuses on the Smart Learning Recommender System which utilizes a novel knowledge-based filtering approach. The algorithm transfers multi-contextual activity data into time-dependent user models. The resulting relevance scores represent the individual user's need to learn specific learning objects at a particular point in time of the course. In comparison to the evaluation results of other educational recommender systems on the same datasets, the introduced approach produces the most precise time-sensitive recommendations.

Keywords

Educational Recommender Systems, User Modeling, Knowledge-based Filtering, Time-Dependent Recommendations

1. INTRODUCTION & RELATED WORK

Recommender Systems are designed for users who lose track of the vast quantity of items available and thus need assistance with their item selection [14]. This paper presents a recommender systems (RSs) for educational closed-corpus settings, which means that the learning environment adapts to the learners' needs within a self-contained course [13]. The user interface offers content recommendations at particular points in time that fit with the current needs and the contextual situation of individual course participants. The most important task of the recommender system is to support learners in achieving their personal learning goals. An

"appropriate" recommendation in this context aims at making learning more efficient and effective by supporting the content selection process [15].

Most researchers borrow traditional Content-based Filtering and Collaborative Filtering algorithms and, at least partially, adapt them to the domain of Technology Enhanced Learning instead of directly applying the same approaches [3]. Bauman and Tuzhilin [1] categorized educational recommender systems either as "knowledge enhancing" which aims at broadening the knowledge to new topics or as "remedial" which aims at filling identified knowledge gaps of previously studied items. Closed-course RSs, such as the one introduced in this work, aim at solving both issues. They assist the learners in reaching their personal or course-specific goals which require new topics to be learned and the consolidation of already acquired knowledge. An exhaustive literature review on recommender approaches for curriculum planning and the prediction of appropriate item sequences in courses is presented in [7].

The remainder of the paper is structured as follows: Section two introduces the overall Smart Learning Recommender with the time-dependent learning need and the Top-N scoring, followed by the determination procedure for optimal factor weights. Section four presents the results of different evaluations. Finally, this paper ends with a description of its limitations and a conclusion.

2. SMART LEARNING RECOMMENDER

The Smart Learning Recommender (SLR) presented in this paper is utilized by the Smart Learning Companion App [8] – a mobile web application¹ that was used by about 600 learners from different institutions so far (until February 2019). Thereby, the web app is the only entry point for students to access courses, learning objects, and lecture dates as well as to get recommendations for the next best contents to be learned and triggers the tracking of all relevant user interactions. A course is usually structured in learning units which themselves contain low-level learning objects (LO) that can be texts, exercises, videos and so on. The hierarchy however can have more levels. The content creators and teachers provide a lot of metadata on each low-level LO based on the Learning Resource Meta-data Specification (LOM), the

¹Smart Learning Companion Application:
<https://smartlearning.fokus.fraunhofer.de/>

Christopher Krauss, Agathe Merceron and Stefan Arbanowski
"Smart Learning Object Recommendations based on Time-Dependent Learning Need Models" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 599 - 602

Question and Test Interoperability Specification (QTI) and the Common Cartridge Specification (CC). Among others, the metadata include attributes on prerequisite LOs, information on the exam relevance of an item, on the typical learning time and on the learning objectives. Interactions with all items are stored as Experience API (xAPI) statements.

The educational recommender system aims at identifying the learning need of a user u for an item i . The learning need is a score that takes the currently approximated knowledge and other related properties (time of the lecture, prerequisites, etc.) as well as the required amount of knowledge into account. The user-item-pair is presented by a relevance score $rscore_{u,i}$ having the value from 0 to 1, where 0 indicates the lowest relevance and 1 indicates the highest possible relevance. The relevance score defines a time and context dependent value and is expressed as a time dependent function: The relevance function $rf_{u,i}(t)$ of user u for item i is derived from several sub-functions $rf_{u,i,x}(t)$ of individual factors x_1, \dots, x_n , as a function of time t , each representing another context (cf. [6]). A detailed description of the factors, including the mathematics, can be found in [7]. All factors are in the range of [0,1] and represent the learning need. The considered factors per user, item and time are:

1. **Self-assessments:** A student can explicitly define his / her knowledge level at particular points in time on a 1 to 5 stars scale
2. **Interaction with items:** This factor indicates how much of the available material for a learning object was accessed by a student
3. **Processing time of an item:** This factor indicates how long the student learned a learning object. It is 0 when the student needed exactly the intended time and between 0 and 1 if he /she needs more or less time as defined in the metadata
4. **Performance in exercises:** The percentage of wrong answered questions represents the relevance of the exercise factor
5. **Fulfilled prerequisites:** The more a student learned the underlying learning objects, the higher the relevance score of the subsequent items
6. **Lecture times:** The lecture times factor indicates the timely relevance of a learning object for face-to-face lectures
7. **Item relevance for the course goal:** learning objects that are more relevant for exams show a higher relevance score than optional contents – in terms of a constant value defined in the Learning Resource Metadata Specification.
8. **Collaborative learning need:** The relevance functions of similar users on this learning object are taken into account in order to offset underestimations and bad learning plannings for the current user.
9. **Human memory and forgetting effect:** After learning an object, the gained knowledge will slowly decrease over time. After each learning iteration, the forgetting factor is set to 0 and then slowly increases again.

For the calculation of recommendations, all single-factor functions are weighted. The weighted average of all factors describes the total learning need of item i for that user u and is calculated as

$$rf_{u,i}(t) = \frac{\sum_{x=1}^n (w_x * rf_{u,i,x}(t))}{\sum_{x=1}^n w_x} \quad (1)$$

Here w_x is the weight of a single factor x in $\{x_1, \dots, x_n\}$ and n is the number of factors (cf. [6]). The overall recommendation engine analyses the current learning need values of the requesting student for all items in the course. Thereby, the model of the SLR can be created offline: The relevance functions are computed at regular intervals by processing all existing user-item-time-triplets. When a user requests recommendations, the relevance scores for all items are calculated on demand by considering the current time value for t . Afterward, the items are sorted by their learning need value. The result is a Top-N list of items beginning with the highest relevance scores that represent the most important learning units or learning objects for the student that needs to be learned at that time.

3. TIME-DEPENDENT WEIGHTS

In order to evaluate the ideal weights for different factors at various points in time that will lead to the most appropriate recommendations, different methods have been analyzed – from equal weights, over manual adjustments, up to linear models that calculate appropriate weights per course week. At the beginning of the development of the Smart Learning Recommender, the weights were set to a value of 1 in order to have a starting point from which to present recommendations to learners. The method that generates the weights which leads to the highest precision values is realized with the help of an Artificial Neural Network (ANN). RapidMiner² is used with a neural network for regression problems that contains just one hidden layer. One input vector represents a user-item-time triplet. The input vector comprises $n + 1$ input features which are the n factor relevance values of the SLR (per user, item and time). Additionally, the first input value per node is set to a value of one, which makes it a node-specific bias. The node's output o_k is the predicted value (e.g., the overall learning need) for the user-item-time triplet. The training values for the output are 1 if this item i has been accessed by the user u in the last week before the split t and 0 if not. The node's output is calculated as:

$$o_k = \varphi\left(\left(\sum_{x=1}^n rscore_{u,i,t,x} * nw_{x,k}\right) + b_k\right). \quad (2)$$

In contrast to the actual intention of an Artificial Neural Network, where the predicted output values o_k are of interest, the approach in this work targets the determined weights of the trained nodes. Thereby, each of the nodes contains a weight vector \overrightarrow{NW}_k . This weight vector has the same number $n + 1$ of dimensions as the ANN has as input features (where the first weight represents the bias b_k). The activation function φ is a sigmoid function which is common for ANNs that aim at solving linear problems. The resulting weights fit the requirements of the SLR weighting formula presented in Formula 1. If the hidden layer contains more than one node, the weights per factor x are averaged according to the node's bias b_k which results in a final weight w_x

²RapidMiner. See: <https://rapidminer.com/>

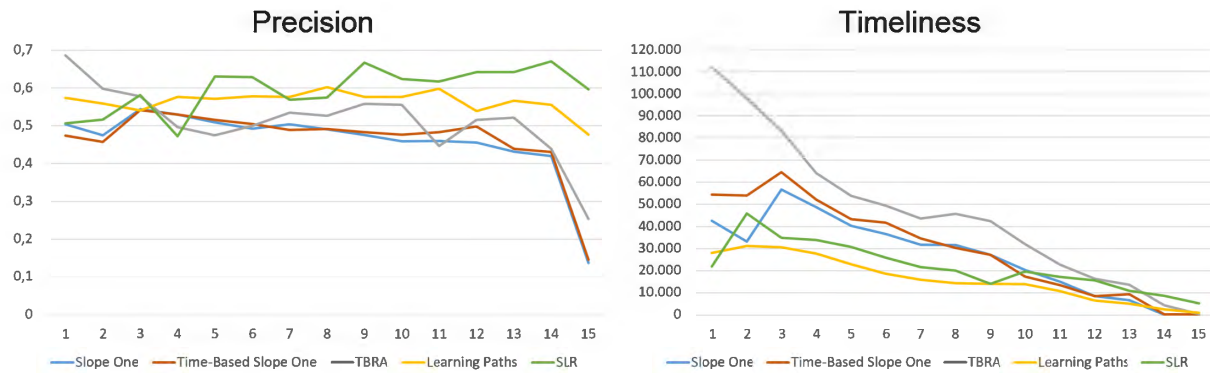


Figure 1: Comparison of the evaluated approaches including the Smart Learning Recommender (each given in the best setting); left: Average precision per course week; right: Timeliness values given in minutes per course week.

of factor x :

$$w_x = \frac{\sum_{k=1}^K n w_{x,k} * b_k}{\sum_{k=1}^K b_k}. \quad (3)$$

K is the number of all nodes in the hidden layer. The determined weight w_x for each factor is subsequently applied to the factor weight algorithm of the Smart Learning Recommender given in Formula 1. An exhaustive description of the approach is presented in [7]. With a sample of 50 nodes, the ANN reaches the highest average precision that is 0.4% better than the precision of other linear models and even 10.9% more precise than the utilization of equal weights.

4. EVALUATION

We compared the Smart Learning Recommender approach to four other approaches. The first recommender system is the basic Slope One algorithm as Collaborative Filtering (CF)-baseline as introduced by Lemire et al. [12]. Lemire et al. considered this approach only theoretically for Technology Enhanced Learning (TEL) [11] and it has only been rarely applied in education contexts as mentioned by Verbert et al. [16]. The second algorithm extends the Slope One approach with time-weights [5]. This time-based algorithm is applied in the context of Technology Enhanced Learning for the first time. Another Item-based Collaborative Filtering approach, the Time-based Recommender Approach for Lecture Materials (TBRA) [4], is based on time-dependent item similarities for the recommendation of study materials. The actual algorithm of presenting similar items has been adopted in this work to fit the recommender's goal and the evaluation approach. The fourth algorithm is, besides the SLR also a novel one. It generates personalized learning paths based on the activities of classmates and the previous interactions of the concerned learner [10]. The next items on the predicted learning path are, therefore, considered as Top-N recommendations.

For training and testing, we used activity data that we collected in a particular face-to-face university course with 99 students. The self-designed learning web app gave access to the course materials which comprised 1,006 learning objects grouped into 106 learning units (for more details see [8]). Thereby, a single learning object has a typical learning time

of, at the most, five minutes. A learning unit groups ten learning object (LO) items on average. We collected 44,421 xAPI statements which represent the item accesses of the 99 students. Instead of a traditional n-fold cross-validation, we used an extension of the "increasing time-window" cross-validation of Campos et al. [2] and a new measurement value, the timeliness [9], that presents the average timespan when a recommended relevant item has been accessed after its recommendation. In contrast to precision and recall, the timeliness measure must be as low as possible.

In the context of all evaluated recommender systems, the Smart Learning Recommender performs best on average regarding precision. Precision is the fraction of relevant items among the items of the Top-N list. Figure 1 visualizes the precision and timeliness of all evaluated approaches – each given in the best setting for the Top-3 recommendations for the university course. As seen on the precision chart on a weekly basis, the SLR (green line) is more scattered than the Learning Path algorithm that shows a worse but more stable trend. Regarding timeliness, the Learning Path algorithm still outperforms the Smart Learning Recommender. However, the Smart Learning Recommender also shows low timeliness values that are the second best on average which means that the recommendations are also very time accurate.

In a second experiment, we analyzed the effect of different learning patterns on the precision: Thereby, more precise recommendations have been obtained using only the most 33 most active learners, and not – as initially expected – the most successful learners (who completed the exercises most successfully on average). The SLR algorithm reaches a precision of 0.800 on average (compared to the previous value of 0.583 with all students). Of course, this is only an artificial setting, because the SLR cannot forecast the most active users at an early point in time and, more importantly, the recommender system should provide recommendations for all users – not only for the most active or most successful users. However, this indicates the potential of the algorithm, as the Smart Learning Recommender aims at recommending items for all users based on favorable patterns. These patterns might be, for instance, the activity patterns of the most active users. We also applied the SLR algorithm on activity data of two other courses – an optional online-

only course with 28 learners at another university and an adult education blended learning course that regularly reoccurs with eight craftsmen each. All courses show an effect of the cold-start phase with lower precision values at the beginning. On average, the two new courses reach almost similar high precision values of 0.818 (for the online-only course) and 0.815 (for the craftsmen's course). This confirms the findings of Verbert et al. [16] that the precision results of educational recommender systems highly depend on the dataset selection and, thus, implicitly on the course setting and the course participants.

5. LIMITATIONS & CONCLUSION

The Smart Learning Recommender introduces a relevance score that represents a numerical value indicating the need for learning (implicitly the knowledge gap) of a learner. This learning need is modeled through different factors and the resulting recommendations are presented in particular categories – such as "Exercises where you received weak results", "Things you might have forgotten" or "Please wrap-up these learning objects". The weights of the context factors are not static. The SLR shows that precision can be increased dramatically when modeling context factor weights over time. Thereby, the weights are determined on a weekly interval to understand the best composition. A dynamic weighting at every point in time allows for the generation of the most precise Top-N list. Due to the evaluation settings, time weights have only been analyzed at weekly intervals. We determined the most appropriate weights based on the most recent activities of the current user. Since the algorithm can also assign a weight of 0 to the factors, it is not counterproductive to consider as many factors as possible. However, other factors that additionally provide useful information about the need for learning still need to be investigated in the future.

The evaluations have been conducted in a simulation environment – with historical real-world data but in the absence of presenting the recommendations of all evaluated algorithms to real-world learners. Thus, the effect of the user interface and its usability have not been compared for the different approaches. Since these aspects have a huge impact on general user acceptance and user perception of the system as a whole, they require further experiments. While the different algorithms were evaluated with the help of a novel evaluation framework, the results can only indicate their various levels of appropriateness for the self-collected activity data. In general, the approaches should work similarly well on other datasets that offer the same kind of user, item and activity information. However, this has not yet been investigated.

6. REFERENCES

- [1] K. Bauman and A. Tuzhilin. Recommending remedial learning materials to the students by filling their knowledge gaps. In *Proceedings of the EdRecSys16*, 2016.
- [2] P. G. Campos, F. Díez, and I. Cantador. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 24(1-2):67–119, 2014.
- [3] M. A. Chatti, S. Dakova, H. Thus, and U. Schroeder. Tag-based collaborative filtering recommendation in personal learning environments. *IEEE Transactions on Learning Technologies*, 6(4):337–349, 2013.
- [4] C. Hermann. Time-based recommendations for lecture materials. In *Proceedings of the 2010 EdMedia*, pages 1028–1033, 2010.
- [5] T. Q. Jiang and W. Lu. Improved slope one algorithm based on time weight. In *Instruments, Measurement, Electronics and Information Engineering*, volume 347 of *Applied Mechanics and Materials*, pages 2365–2368. Trans Tech Publications, 10 2013.
- [6] C. Krauss. Smart learning: Time-dependent context-aware learning object recommendations. *Proceedings of The 29th AAAI FLAIRS Conference*, May 2016.
- [7] C. Krauss. *Time-Dependent Recommender Systems for the Prediction of Appropriate Learning Objects*. Dissertation at the Technical University Berlin, <http://dx.doi.org/10.14279/depositonce-7119>, 2018.
- [8] C. Krauss, A. Merceron, T.-S. An, M. Zwicklbauer, S. Steglich, and S. Arbanowski. Teaching advanced web technologies with a mobile learning companion application. In *Proceedings of The 16th ACM mLearn Conference, ACM, Larnaca, Cyprus*, 2017.
- [9] C. Krauss, A. Merceron, and S. Arbanowski. The timeliness deviation: A novel approach to evaluate educational recommender systems for closed-courses. In *Proceedings of The 9th LAK Conference, LAK19*, pages 195–204, New York, NY, USA, 2019. ACM.
- [10] C. Krauss, A. Salzmann, and A. Merceron. Branched learning paths for the recommendation of personalized sequences of course items. *Proceedings of the Learning Analytics Workshop, co-located with the 16th e-Learning Conference of the German Society for Computer Science, Frankfurt, Germany, September 10, 2018*.
- [11] D. Lemire, H. Boley, S. McGrath, and M. Ball. Collaborative filtering and inference rules for context-aware learning object recommendation. *Interactive Technology and Smart Education, Emerald Group Publishing Limited*, 2(3):179–188, 2005.
- [12] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. *Proceedings of The SDM'05 Conference*, 2005.
- [13] N. Manouselis, H. Drachsler, R. Vuorikari, H. Hummel, and R. Koper. Recommender systems in technology enhanced learning. In *Recommender systems handbook*, pages 387–415. Springer, 2011.
- [14] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167. ACM, 2000.
- [15] N. Tintarev and J. Masthoff. Designing and evaluating explanations for recommender systems. In *Recommender Systems Handbook*, pages 479–510. Springer, 2011.
- [16] K. Verbert, H. Drachsler, N. Manouselis, M. Wolpers, R. Vuorikari, and E. Duval. Dataset-driven research for improving recommender systems for learning. In *Proceedings of The 1st Conference LAK*, pages 44–53. ACM, 2011.

Teacher vs. Algorithm: Double-blind experiment of content sequencing in mathematics

Ben Levy
Ben Gurion University, ISRAEL
levybeny@post.bgu.ac.il

Arnon HersHKovitz
Tel Aviv University, ISRAEL
arnonhe@tauex.tau.ac.il

Odelia Tzayada
Tel Aviv University, ISRAEL
odelya@mail.tau.ac.il

Orit Ezra
Tel Aviv University, ISRAEL
oritezra@mail.tau.ac.il

Avi Segal
Ben Gurion University, ISRAEL
avisegal@gmail.com

Kobi Gal
Ben Gurion University, ISRAEL
University of Edinburgh, U.K.
kobig@bgu.ac.il

Anat Cohen
Tel Aviv University, ISRAEL
anatco@tauex.tau.ac.il

Michal Tabach
Tel Aviv University, ISRAEL
tabachm@tauex.tau.ac.il

ABSTRACT

We study content recommendation in an online learning environment for mathematics ($N=77$, 4th-5th grade student). We compare an expert teacher's recommendation to that of a neural network algorithm, implementing collaborative filtering ranking. We do so using a double-blind randomized controlled experiment. We find that when the difficulty of the teacher's sequence of recommendation was overall increasing, the teacher was superior to the algorithm regarding students' performance. Taken together, our findings indicate on how the algorithm and the expert teacher can benefit from each other.

Keywords

Content sequencing, neural network, collaborative filtering.

1. INTRODUCTION

Designing a learning process that is characterized by increasing task difficulty is a desired goal, for keeping learners challenged, motivated and engaged. Meeting this goal is also a matter of personalization. In some contexts of computer-based learning—e.g., in intelligent tutoring systems—such a personalization has been in practice for many years [1], [2]. In other e-learning settings, as well as in traditional classrooms, this is still a great challenge.

Personalization requires tailoring future content, based on past engagement with materials, and based on the difficulty of the materials. One of the most common approaches to do so is collaborative filtering [3]. We implement a novel adaptation of collaborative filtering, by ranking questions in order of their subjective "difficulty" to the student, with the latter defined based on different aspects of similar students' behavior while engaging with past questions. This method has proven superiority over other personalization algorithms [4]. Following that, we ask whether an expert teacher can do the same.

Ben Levy, Arnon HersHKovitz, Odelia Tzayada, Orit Ezra, Avi Segal, Kobi Gal, Anat Cohen and Michal Tabach "Teacher vs. algorithm double-blind experiment of content sequencing in mathematics" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 603 - 606

In order to conduct a fair comparison between human and computer recommendations, we used a double-blind randomized controlled study (RCT). RCTs have been underused in educational research [5]. Despite a recent boom in its use in computer-based educational settings [6], conducting double-blind RCTs in authentic classroom environments is still a rarely-met challenge.

2. METHODS

2.1 Population

Participants were 77 students in 4th-5th grades from a public school in a metropolitan area in Israel, 2 classes in each grade-level. In 4th-grade, there were 22 boys, 24 girls; in 5th-grade – 12 boys, 19 girls.

The expert teacher (third author) teaches mathematics in that school. She has 18 years of experience in teaching and about a decade of experience in incorporating technology in classroom. She is also well-familiar with the learning environment used here.

2.2 The Learning Environment System

We used a commercial online learning environment which includes game-based applets for K-12 school mathematics (per the request of the company, we do not mention its name). Each applet—consisting of a few problems that each allows multiple attempts—is aimed at a specific sub-content, usually focusing on a specific skill (e.g., identify fractions, make fractions, name fractions, compare fractions, etc.). Applets usually require some interaction from the learner (e.g., dragging and dropping visual elements) and give them feedback upon submitting a solution. We focus on the topic of fractions.

2.3 Data, Definition of Difficulty

We computed the absolute measure of difficulty for each applet based on historic data collected from students using the online learning environment. As each student can interact more than once with each applet, we used only the first attempt of each student with each applet. After preprocessing and cleansing of the data, each student-applet interaction contains the following tuple: <student id, applet id, score, start time, time spent on applet>. This dataset includes 563,735 rows, documenting 15,000 Israeli students who used 978 applets during 9/17-6/18.

For calculating the absolute difficulty of the applets used for our intervention, we sorted them based on their average score (according to the historic data), and used average time spent as a tiebreaker. Based on this ranking, we then divided the applets to

three equally-sized groups: levels 1 (easiest) to level 3 (most difficult). We did so for each grade-level separately.

2.4 Research Process

Data collection was done during December 2018-January 2019.

2.4.1 Partition to Research Groups

Partition to Algorithm/Teacher groups was based on a pen-and-pencil pre-test scores. Students were ordered by their score, and were assigned to Algorithm/Teacher groups, alternately.

2.4.2 Phase I Data Collection (Baseline)

During phase I, which served as a baseline data collection, all students of the same grade-level were assigned with the same set of 10 applets, in the same order. These applets were chosen by the expert teacher, based on the class syllabus. The students worked individually at the school's computer lab, with the teacher and part of the research team present but not intervening or helping them at any point. Students had about 90 minutes to complete the full set, and most students indeed did so; those who did not finish on time, came to the lab at a later time to complete the set.

The Teacher and Algorithm groups demonstrated similar behavior in Phase I activity, when measuring percentage of problems correctly answered on the first complete run of the applet and the time spent on the first complete run of the applet (see Table 1).

Table 1. Comparing the research groups' activity in Phase I

Grade	Group	Mean (SD) % Corr. Prob.	Mean (SD) Time Spent
4	Algo. (N=21)	0.76 (0.15)	210.47 (48.60)
	Teacher (N=25)	0.74 (0.20)	235.26 (94.33)
t-value		t(44)=0.45 [‡]	t(37.12)=1.15 [‡]
5	Algo. (N=15)	0.79 (0.22)	149.20 (41.50)
	Teacher (N=16)	0.88 (0.17)	142.36 (33.73)
t-value		t(29)=1.21 [‡]	t(29)=0.51 [‡]

[†] Levene's Test was significant [‡] Not significant

2.4.3 Constructing the Recommendations

Log files from Phase I were collected, preprocessed and analyzed. The performance of the students in Phase I served as input to the algorithm and teacher for selecting the applets for phase II. Both the computer and the teacher were given the same pool of pre-chosen applets (12 for grade 4, 20 for grade 5) from which they could assign 10 applets for Phase II. This set was focused on continuing learning and practicing fractions.

Both the teacher and the algorithm produced their recommendations for the full cohort of the participants. In order to ensure the teacher's "blindness", she was not allowed—at any stage of the experiment (in-class activities, report-examining, or recommendation-making)—to connect students with their system id; she was also not involved in the pre/post-test writing or grading.

Algorithm's Recommendation. We base our personalized recommendation on the NCFR algorithm output (Section 3.2) and then adapt it to our Phase II pool of applets. For each student, we place a sliding window in the middle of her or his ranking, and move this window one applet towards the less/more difficult applets, according to the their Phase I performance (one step for each SD below/above average of their grade-level, respectively).

Teacher's Recommendation. The teacher was presented with a table summarizing score and time spent for each student on each applet. Examining it, she decided to divide the students (in each

grade-level) to three level-groups ("weak", "intermediate", "strong"), and to assign a different sequence of 10 applets to each.

2.4.4 Phase II Data Collection (Intervention)

Each student was assigned with the applets they were recommended with, in accordance with their research group. Working conditions for students were the same as in Phase I.

2.4.5 Post-test

A few days after Phase II, the students took a pen-and-paper post-test on the material covered by Phase II. We observed no differences in post-test scores between the two groups. In grade 4, the average score (on a scale of 0-26) for the Algorithm group was 18.48 (SD=6.82) and for the Teacher group – 18.28 (SD=6.42), with t(44)=0.1, at p=0.92. In grade 5 (on a scale of 0-25), the average score for the Algorithm group was 17.00 (SD=7.52) and for the Teacher group – 18.06 (SD=6.27), with t(29)=0.43, at p=0.67.

3. RECOMMENDATION ALGORITHM

3.1 Problem Definition

In this section we define the *relative* measure of difficulty used by the algorithm to personalize applets to students. We consider an e-learning setting with a group of students S and a set of questions Q . The "difficulty ranking problem" includes a target student $s_i \in S$, and a set of questions $T_i \subset Q$, for which the algorithm must predict a difficulty ranking over T_i .

The input to the problem includes: (1) A set of students S ; (2) A set of questions Q ; (3) For each student $s_j \in S$, a partial ranking by difficulty over a set of questions $H_j \subset Q$. We assume a personal difficulty ranking over questions for each student. We compute the personal difficulty ranking by ranking questions according to students' first attempt score, breaking ties by time spent.

For every student $s_j \in S$ there are two disjoint subsets $H_j, T_j \subset Q$, where the difficulty ranking of s_j over H_j is known. For a target student $s_i \in S$, H_i represents the set of questions that the target student s_i has already answered (i.e., student's history), while T_i is the set of questions for which a difficulty ranking is needed (i.e., target questions). The task is to leverage the known rankings of all students s_j over H_j in order to compute the required difficulty ranking over T_i for student s_i .

In order to preserve the temporal order of events, we split the data to train and test sets; for each student, we sort his/her interactions by start time, then split the sorted interactions by taking the first 70% of the interactions as the train set and the remaining 30% of interactions as the testing test set.

3.2 Neural Collaborative Filtering Ranking

Our Neural Collaborative Filtering Ranking (NCFR) approach is a hybrid model based on two different fields: collaborative filtering and information retrieval. We apply the multi-layer neural network representation detailed in [7] to our domain. Figure 1 shows the network architecture, where the input is two one-hot encoded vectors that index student-question identity. These vectors are fed into two separate, fully connected embedding layers that output dense vectors which are a latent representation for the student and question (similar to latent factor models in collaborative filtering). The dense vectors are concatenated and fed into a multi-layer, fully connected neural architecture, mapping the latent vectors to a difficulty prediction score that depends on both student and question. The final output layer is the predicted difficulty score \hat{d} .

The training process of our model, Learning to Rank (LTR), is adapted from similar tasks in information retrieval [8]. Our network optimizes a LTR pairwise loss function, predicting the subjective difficulty of one question over the other for each given student. A forward pass of the network computes the probability that question q_i is more difficult than question q_j for student s . We separately predict the difficulty score $\hat{d}(s, q_i)$ and $\hat{d}(s, q_j)$, using the network. The probability that question q_i is harder than question q_j for student s is: $\hat{y}(s, q_i, q_j) = \sigma(\hat{d}(s, q_i) - \hat{d}(s, q_j))$.

Where σ is a *sigmoid function*. For example, when the score for q_j and q_i is the same, the model is fully agnostic (each question is harder with probability 0.5).

The objective function to minimize for NCFR is the *binary-cross-entropy*, and its optimization can be done by performing stochastic gradient descent (SGD):

$$L = -(I(s, q_i, q_j) * \log \hat{y}(s, q_i, q_j) + (1 - I(s, q_i, q_j)) * \log(1 - \hat{y}(s, q_i, q_j)))$$

Where $I(s, q_i, q_j)$ (i.e., the indicator function) is 1 when question q_i is more difficult than question q_j for student s , and 0 otherwise.

We use Copeland's pairwise aggregation method [9] to convert the pairwise difficulty predictions for all students to a ranking over the target question set, using the same procedure as Segal et al. [4].

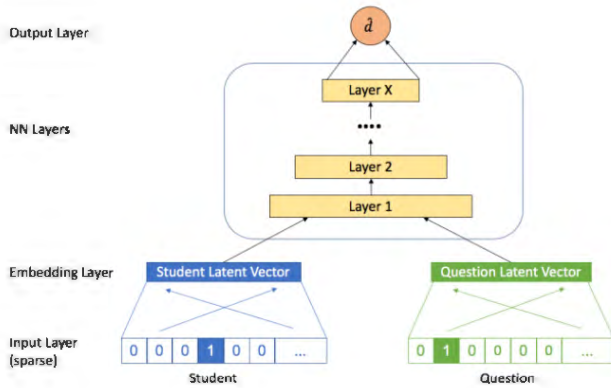


Figure 1. Neural network architecture (adapted from [7])

4. FINDINGS

4.1 Overall Difficulty in Phase II

We examine the difficulty of the teacher's assignment to "weak", "intermediate" and "strong" group of students based on the data collected in Phase I. Calculating averages of the teacher's recommended applets for each group, we get an increasing value from the "weak" to the "intermediate" and from the "intermediate" to the "strong" groups. (As we compare averages of 10 numbers in each case, we do not run statistical analyses for significance.) This indicates an overall **alignment between the teacher's evaluations of the applets' difficulty level and their absolute difficulty measure**. Findings are summarized in Table 2.

Table 2. Average difficulty of teacher's group-based sequences

Grade	"Weak"	"Intermediate"	"Strong"
4	1.9	2.1	2.4
5	1.4	1.8	2.5

We now compare between the teacher's and the algorithm's recommendations' difficulty for Phase II. We calculate for each student the average of difficulty across their 10 recommended

applets, then calculate the average across students. Overall, the **teacher, compared with the algorithm, recommended more difficult applets**. These differences are statistically significant, although denote a small effect size (see Table 3).

Table 3. Comparing difficulty of recommendations

Grade	Avg. (SD) Teacher	Avg. (SD) Algorithm	Paired t-test	Effect Size [†]
4 (N=46)	2.17 (0.21)	1.96 (0.11)	9.32***	0.05
5 (N=31)	2.02 (0.44)	1.77 (0.18)	3.84**	0.13

** p<0.01, *** p<0.001, [†] Cohen's d

4.2 Difficulty along Phase II

For comparing between recommended sequences that may differ from each other by their applets and/or by the applets' order, we refer to each student's 10 ordered learning opportunities (LOs). This approach has been repeatedly taken in the EDM community.

We examine the difficulty of the LOs by averaging difficulty for each LO separately (across students), comparing between the teacher's and the algorithm's recommendations. For grade 4, the algorithm's recommendation is characterized by an overall increased difficulty level along the LOs, while the teacher's recommendation has an overall opposite trend. This is supported by calculations of the linear trend for both vectors, with $m=0.13$ for the algorithm, and $m=-0.13$ for the teacher.

For grade 5, we observe an overall increasing trend in both cases; linear trend calculations result with $m=0.15$ and $m=0.07$ for the algorithm and the teacher, respectively. See Figure 2.

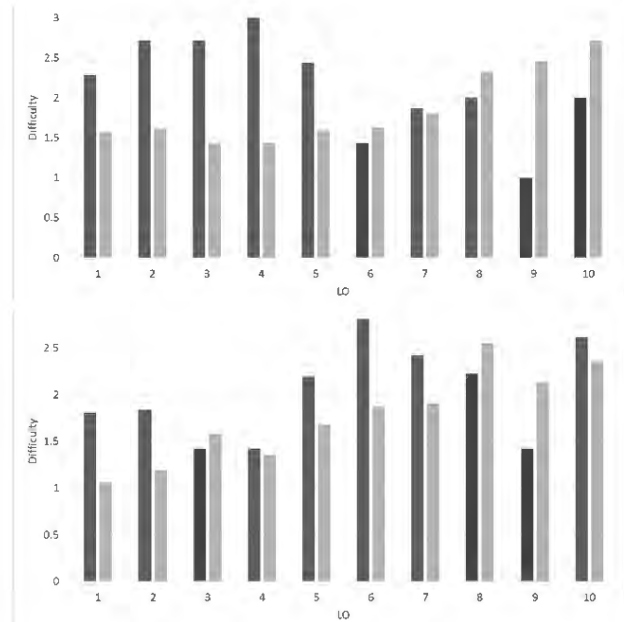


Figure 2. Phase II Recommendations' difficulty (grade 4 – top, grade 5 - bottom), teacher (dark) and algorithm (light)

We also evaluate level of agreement between the teacher's and the algorithm's actual recommendations, using AP correlation [10], which measures similarity between two different orders; we apply this measure for the overlapping of each student's two recommendation sets. For grade 4, the average AP correlation is 0.42 (SD=0.12, N=46). We also note that the average size of the overlapping subset is 6.22 (SD=0.87). For grade 5, the average AP correlation is 0.75 (SD=0.22, N=31), with an average overlapping

subset size of 5.19 (SD=0.79). These are in line with the abovementioned findings: where both recommendations demonstrate increasingly difficulty (grade 5), we observe high alignment between the recommended sets, while where there are opposite trends (grade 4), the level of alignment is much lower.

4.3 Students' Actual Behavior

For each LO, we calculate the average score across each research group. As may be seen in Figure 3, in grade 4 there is no clear advantage to either the teacher (N=25) or the algorithm (N=21) groups, while in grade 5, there is an overall advantage for the teacher group (N=16) over the algorithm group (N=15). Considering the findings reported in the previous section, we may infer that **when the algorithm's and the teacher's recommendations align – the teacher has an advantage over the algorithm; in contrary, where the algorithm's and the teacher's recommendations do not align – neither the teacher nor the algorithm demonstrates an advantage over the other.** Note that due to small sample sizes, we do not run statistical analyses here.

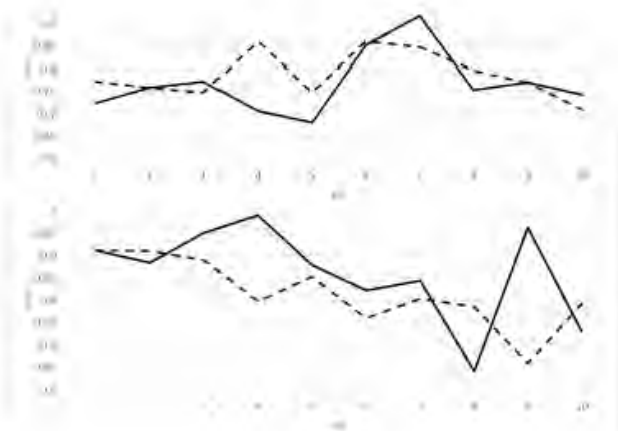


Figure 3. Phase II scores for grades 4 (top), 5 (bottom), comparing teacher (solid) and algorithm (dashed) groups

5. DISCUSSION

In this study, we compared between difficulty level of personalized recommendations of an expert teacher and a novel algorithm, in the context of elementary school mathematics. We find that the teacher had overall recommended more difficult applets than the algorithm. However, students in the teacher group did not achieve lower than their peers in the algorithm group. Therefore, we may suggest an improvement to the algorithm—and educational software at large—i.e., enabling more difficult tasks, in order to maximize students' engagement and learning within their zone of proximal development [11].

Additionally, we show that the teacher did not always recommend an increasingly difficult sequence of applets, while the algorithm—by its very nature—did so indeed. When such a trend was observed in the teacher's recommendations (grade 5), the teacher demonstrated superiority over the algorithm regarding students' performance during intervention (Phase II). This may be a result of the teacher considering additional parameters to the ones considered by the algorithm; for example, familiarity with the topics and with the content of the applets (it is our plan to interview the teacher regarding her thought process while personalizing applets). This, again, may be taken as a point of improvement to the algorithm. On the other hand, this finding highlights the superiority of objective, data-based difficulty level of the applets, which

probably incorporates—"unconsciously", though this term is not appropriate for a machine—other applet characteristics, such as animation, interactivity, narrative, etc. Taken together, these findings indicate on how the algorithm can benefit from the expert teacher's experience, and vice versa.

The differences in the teacher's recommendations for grade 4 and 5 could be explained by the different pool sizes from which she could choose applets for the intervention phase (the pool for grade 4 was smaller than that of grade 5). We plan on replicating this study with larger pools, as well as with other grade-levels and more topics.

6. ACKNOWLEDGMENTS

This study was supported in part by the Israeli Ministry of Education (Program 35/12.15, "Promoting Technologies for Domain-Specific Teaching").

7. REFERENCES

- [1] T. McGinnis, D. W. Bustard, M. Black, and D. Charles, "Enhancing e-learning engagement using design patterns from computer games," in *First International Conference on Advances in Computer-Human Interaction*, 2008, pp. 124–130.
- [2] D. S. McNamara, G. T. Jackson, and A. Graesser, "Intelligent tutoring and games (ITaG)," in *14th International Conference on Artificial Intelligence in Education Workshops Proceedings*, 2009, pp. 1–10.
- [3] S. Shishehchi, S. Y. Banihashem, N. A. M. Zin, and S. A. M. Noah, "Review of personalized recommendation techniques for learners in e-learning systems," in *2011 International Conference on Semantic Technology and Information Retrieval*, 2011, pp. 277–281.
- [4] A. Segal, Z. Katzir, K. Gal, G. Shani, and B. Shapira, "EduRank: A collaborative filtering approach to personalization in e-learning," in *Proceedings of the 7th International Conference on Educational Data Mining*, 2014, pp. 68–75.
- [5] A. Bouguen and M. Gurgand, "Randomized controlled experiments in education (EENE analytical report No. 11)," 2012.
- [6] J. C. Stamper, D. Lomas, D. Ching, S. Ritter, K. R. Koedinger, and J. Steinhart, "The rise of the super experiment," in *International Conference on Educational Data Mining*, 2012, pp. 196–199.
- [7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web - WWW '17*, 2017, pp. 173–182.
- [8] M. Deeds *et al.*, "Learning to rank using gradient descent," in *Proceedings of the 22nd International Conference on Machine Learning*, 2005, pp. 89–96.
- [9] A. H. Copeland, "A reasonable social welfare function." University of Michigan (Ann Arbor, MI), 1951.
- [10] E. Yilmaz, J. A. Aslam, and S. Robertson, "A new rank correlation coefficient for information retrieval," in *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*, 2016, pp. 587–594.
- [11] R. Luckin, "Designing children's software to ensure productive interactivity through collaboration in the zone of proximal development (ZPD)," *Inf. Technol. Child. Educ. Annu.*, vol. 2001, no. 1, pp. 57–85, 2001.

Studying Factors Influencing the Prediction of Student STEM and Non-STEM Career Choice

Varun Mandalapu
Department of Information Systems
University of Maryland Baltimore County
Baltimore, Maryland, USA 21250
varunm1@umbc.edu

Jiaqi Gong
Department of Information Systems
University of Maryland Baltimore County
Baltimore, Maryland, USA 21250
jgong@umbc.edu

ABSTRACT

The increasing capabilities of intelligent tutoring systems (ITS) that collect student interaction data while learning mathematics at the middle school level have enabled researchers in educational data mining (EDM) to develop models that predict student career choice. Current research focuses on feature selection techniques that provide essential features in predicting the target variable. However, the factors that affect the prediction performance of an algorithm at a sample level could be studied in depth as they influence the overall performance of an algorithm. In this study, we analyze the influence of various attributes collected by the ASSISTments online learning platform on the performance of machine learning algorithms in predicting student career fields. Initially, we adopt a feature selection technique based on correlation, ID-ness, stability, and Missing values to determine useful attributes and then apply machine learning algorithms to classify student field. The trained models will be used to extract the supporting and contradicting attributes that influence the prediction performance of an algorithm. The results showed that the affect state confused played a significant role in supporting Non-STEM prediction, and boredom played a substantial role in contradicting STEM predictions while gaming the system influences both STEM and Non-STEM predictions. This proposed study facilitates researches in the field of EDM with factors that influence the development of efficient models in predicting STEM and Non-STEM careers.

Keywords

STEM Career, Factor Analysis, Feature Selection, Educational Data Mining, Affect State.

1. INTRODUCTION

Investigating factors that influence student interest in STEM fields at middle school level supports researchers to develop methods that help to focus on areas that empowers their interest in STEM as a career choice. The increasing adaptation of learning technologies like Intelligent tutoring services (ITS) and Massive open online courses (MOOC) at school level supports researches in the field of educational data mining (EDM) to predict student

career choices based on their interaction [6]. With the advancement in the design and capabilities of these systems to collect student interaction data, affect data and knowledge data, different models were developed to understand and predict factors that influence student interest in the STEM field [7]. Social cognitive and career theory (SCCT) show evidence that learning and knowledge pattern at a younger age influences student STEM career [5]. The student interest in mathematics during middle and high school years improve their self-efficacy and performance which can be an influential factor for STEM major enrollment [6,10]. Affective engagement and behavioral models developed earlier showed relationship with choice of majors and college attendance.

An earlier study on predicting student career choice from interaction and affect state data showed a negligible effect of affect state in predicting student career [11]. This study extracts features related to knowledge states based on student problem-solving abilities and skills that were used to predict their fields. Although this study discusses the influence of various predictors on predicting student knowledge states, their approach is to average samples in student log instead of utilizing available comprehensive data. These predictors were then subject to feature engineering and feature selection techniques to incorporate them in algorithm training and testing that improves prediction performance. However, even with the careful selection of predictors following reliable methods the performance of trained algorithms in predicting new student samples is not high. In our study, we incorporate feature engineering and feature selection methods to train and test multiple machine learning algorithms and analyze predictors that support and contradict prediction made by these algorithms. This type of factor analysis will develop an understanding of predictors on classification algorithms.

In this study, we adopt a feature selection technique based on stability, ID-ness, and correlation (Pearson) measures of attributes [4]. We developed three categories (Safe, Moderate and Unsafe) of features based on these measures. Features that fall in the safe and moderate categories were then used to evaluate different machine learning algorithms. The highly supporting and contradicting features for each sample in the dataset were identified based on neighboring attribute weights that utilizes correlation as an identification factor. The local linear relation between attributes is highly influential in prediction compared to the non-linear global relationship [2]. Analysis of features that support and contradict predictions related to STEM, Non-STEM predictions facilitates to understand the importance of each feature on individual class prediction.

Varun Mandalapu and Jiaqi Gong "Studying Factors Influencing the Prediction of Student STEM and Non-STEM Career Choice" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 607 - 610

2. DATASET

This study adopts the ASSISTments dataset provided during Educational Data Mining (EDM) competition in 2017. ASSISTments platform captured US middle school student interaction data from 2004 to 2007 school years [7,9]. This dataset consists of 1709 student's system interaction data. These students were requested to participate in a survey conducted to record post-high school career achievement. This survey provided the career choices of 591 students. In this, 466 students belong to Non-STEM field, and 125 students belong to STEM field.

3. METHODOLOGY

This study mainly focuses on three aspects; initially, we perform feature selection then evaluate machine learning models and extract predictors that support and contradict predictions. Figure 1. shows the complete methodology of this study.

3.1 Feature Selection

Feature selection technique based on correlation, stability, ID-ness and missing value of an attribute was adopted [4]. Correlation in this study considers the linear correlation between attribute and target column. The percentage of ID-ness implies the percentage of different values present in a column. For instance, an attribute with incremental values can have an ID-ness of 100%. Stability is the measure of constant values in an attribute. Stability is zero if there are no similar values where stability is 100 percent if all the values are the same in an attribute. Missing value measure is the percentage of values missing in a attribute. We categorized these attributes into three categories based on the measures mentioned above.

The unsafe category consists of attributes that have more than 70% missing values, or the column is an ID column which is decided based on the ID-ness value, or stability more significant than 90 percent or correlation less than 0.0001 percent or higher than 95 percent. This study removes the attributes from the dataset as they diminish the performance of algorithms. The moderate category consists of attributes that have an ID-ness value of 85%, or correlation less than 0.01%, or correlation more significant than 40%. Attributes that fall in this category will have minimal impact on the predictions. This study included these attributes for analysis. This category consists of attributes that have low ID-ness, the correlation between 0.01 and 40 %, no missing values and stability less than 90%. Attributes in this category profoundly positively impact prediction.

3.2 Model Validation

We adopt the RapidMiner data science platform to train and test chosen predictive models [3]. In this study, we evaluated five machine learning models that differ based on their principles. We chose gradient boosted tree (GBT), Deep neural network (DL), AutoMLP(Multilayer perceptron) random forest (RF) and logistic regression (LR). We discuss model hyperparameters in below

subsection. All the algorithms were evaluated using five-fold cross-validation method on features selected from the above method.

3.2.1 Models and Hyperparameters

1. *Gradient Boosted Tree*: Gradient boosted tree algorithm is a sequential learning algorithm in which a subsequent tree learns from the weak predictors of a previously built tree. The tree adopted in this study has a maximum of 20 trees, maximal tree depth of 20 and a learning rate of 0.1.
2. *Random Forest*: A random forest is an algorithm that works based on ensemble learning principle. This algorithm can combine different models developed based on the bagging method. We obtained optimal settings for this algorithm with a maximum of 100 trees and a maximal depth of 10 per tree.
3. *Logistic Regression*: Logistic regression method is for classification problems as it predicts the probability of each class and classifies based on the probability values. We adopt the standard settings for this model.
4. *AutoMLP*: A multilayer perceptron is a feed-forward neural network that consists of multiple hidden layers in training a neural network. The AutoMLP algorithm can set the optimal learning rate and hidden layers during training. This algorithm works on stochastic optimization and genetic algorithms. This algorithm trains small ensemble methods in parallel with different hyperparameter settings like hidden units and learning rate which are validated to find the best setting.
5. *Deep Neural Network*: A deep neural network is an algorithm that can work with different activation layers, learning rates and optimizers. In this study, we adopt a four-layer (input, hidden_1, hidden_2, and output) fully connected deep learning network that has 250 hidden units in each layer. We set the learning rate at 1.0E-5 and use rectifier activation function. The regularization parameters were auto-adjusted based on the training performance of the algorithm

3.3 Confidence Calculation

In this study, we adopted a confidence-based method that calculates the confidence value ranges between 0 and 1 of student prediction based on the actual and predicted label over all their samples. We extract the cross-validation predictions of each algorithm to calculate the confidence of each student and then label their choice of field as STEM or Non-STEM. If the student prediction confidence over all samples is greater than 50 percent, then the prediction is the same as the actual label. If the student prediction confidence is less than 50 percent, then the prediction is opposite to the actual label of the student.

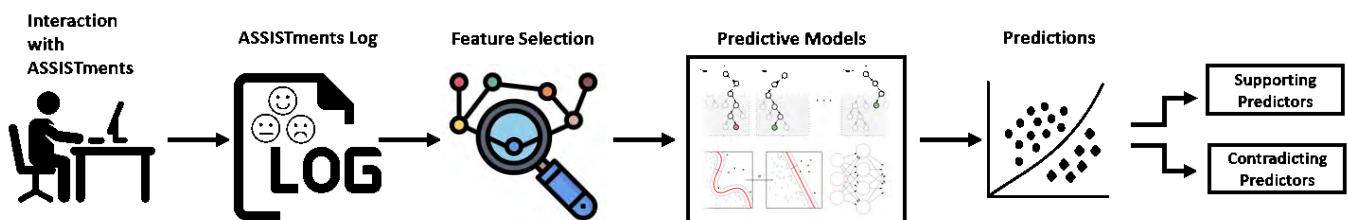


Figure 1: Architectural flow of student career prediction

3.4 Predictor Explanation

This purpose of this study is to understand the factors that influence STEM and Non-STEM predictions. For this purpose, we utilize "explain predictor" operator from RapidMiner to understand the purpose as mentioned above. This method creates neighboring data points for each sample in a dataset and calculates local correlation values to identify the weights of each attribute. The predictors that support and contradict are classified based on the local correlations and weights calculated for each attribute for every sample. The words "supporting" and "contradicting" refer only to the predicted value which might be a accurate or inaccurate prediction. The linear relationship between attribute and prediction locally is highly influential; even the attributes are nonlinear globally.

4. RESULTS

4.1 Cross Validation Performance

This study adopts a cross-validation method to evaluate five machine learning algorithms. Table 1 below shows the cross-validation performance of predictive models built on safe and moderate features categorized by feature selection method. Gradient boosted tree that learns sequentially from weak learners performed better compared to other complex models like deep learning. The performance (AUC, Kappa, and RMSE) of predictive models evaluated on with safe and moderate features show a slight improvement compared to the performance of models based on High impact features.

Table 1: Cross Validation performance of machine learning models on feature selected data with safe and moderate features.

Algorithm	AUC	Accuracy (%)	Kappa	RMSE
Gradient Boosted Tree	0.999	98.83	0.964	0.116+/- 0.002
Deep Learning	0.674	58.19	0.150	0.388 +/- 0.001
AutoMLP	0.623	79.89	0.048	0.396 +/- 0.002
Random Forest	0.635	79.63	0.004	0.398 +/- 0.000
Logistic Regression	0.588	79.58	0	0.400 +/- 0.000

The confusion matrices for STEM and Non-STEM predictions for 591 students were developed based on a confidence cutoff value at 0.5. The below-mentioned table 2 are confusion matrices with Recall and precision scores calculated. As observed earlier GBT and DL does better with high class precision and recall values compared to AutoMLP, RF and LR which were unable to predict STEM classes.

4.2 Explain Predictions

The main focus of this study is to understand the predictions made by the adopted machine learning algorithms. For this purpose, we extract all the supporting and contradicting attributes and present

top six in below Tables 3 and 4 for both accurate and inaccurate predictions. These supporting and contradicting algorithms were classified based on the local Pearson correlation values obtained by calculating the correlation between the attribute and prediction made. One should be careful in interpreting support and contradict predictors. For instance, a supporting predictor for a sample with accurate prediction (Actual Label = Predicted Label) means that this predictor acted positively on predicting actual label whereas a supporting predictor for inaccurate prediction (Actual Label \neq Predicted Label) means that this predictor acted negatively for this prediction. This explanation is similar for contradicting predictors, where the contradicting predictor has negative effect on accurate predictions and positive effect on inaccurate predictions.

Table 2: The below tables shows the confusion matrices with their class recall and precision values for all five machine learning algorithms adopted in this study.

Gradient Boosted Tree	True ST	True NS	Class Precision (%)
Pred. ST	124	0	100
Pred. NS	1	466	99.79
Class Recall (%)	99.20	100	

Deep Learning	True ST	True NS	Class Precision (%)
Pred. ST	95	226	29.60
Pred. NS	29	240	89.21
Class Recall (%)	76.61	51.50	

Table 3: Supporting and Contradicting predictors related to GBT model

Accurate Prediction		Inaccurate Prediction	
Supporting	Contradicting	Supporting	Contradicting
NumActions	sumRight	RES_GAMIN G	NumActions
timeGreater10 SecAndNext ActionRight	totalFrAttempted	totalFrAttempted	timeTaken
original	sumTimePerSkill	frPast8Wrong Count	sumTimePerSkill
frPast5HelpRequest	frPast8Wrong Count	hintCount	sumRight
correct	totalFrSkillOpportunities	totalFrSkillOpportunities	totalFrPastWrongCount
manywrong	RES_GAMIN G	totalTimeByPercentCorrect Forskill	Ln

Table 4: Supporting and Contradicting predictors related to Deep Learning model

Accurate Prediction		Inaccurate Prediction	
Supporting	Contradicting	Supporting	Contradicting
totalTimeByPercentCorrectForskill	NumActions	NumActions	timeTaken
timeTaken	totalFrAttempted	totalFrAttempted	sumRight
endsWithScaffolding	attemptCount	frPast8WrongCount	hintCount
sumRight	totalFrSkillOpportunities	frTotalSkillOpportunitiesScaffolding	endsWithScaffolding
correct	frPast8WrongCount	attemptCount	hint
sumTimePerSkill	frTotalSkillOpportunitiesScaffolding	totalFrSkillOpportunities	frPast5HelpRequest

5. DISCUSSION

This study explores the importance of feature selection and investigates the local linear correlation of predictors on predictions. Affect states, knowledge traces, and clickstream records were studied extensively to understand their impact on model predictions. We observe that the "NumActions" has a high impact on overall accurate predictions of GBT but adversely effects Deep Learning algorithm, this might be due to the differences in the statistical background of algorithms and their regularizations functions. Now in case of accurate STEM prediction made by GBT model, attempts and clickstream records support the prediction whereas affect state boredom and disengaged behavior off-task acts negatively on accurate STEM predictions. Affect state confused has a high positive influence in predicting Non-STEM class and disengaged behavior gaming also supports an accurate prediction of this class. Affects states impact on deep learning algorithm seems to be negligible as most of the predictions depend on knowledge states and clickstream records. Gaming the system has negative impact on STEM career prediction and overall predictions. A previous study by San Pedro et al. also found this relationship between gaming the system and Non-Stem students during their major selection [9]. One reason for the pattern mentioned above might be related to students turning from boredom to off-task which negatively impacts STEM choice [1].

Previous studies suggested a high correlation between carelessness and STEM students [6,8]. In this study, the impact of Average carelessness attribute available in this dataset is investigated to check the model performance based on its presence and absence. With the inclusion of average carelessness, the performance metrics of the GBT model increased. From the predictor explanation, we observe that the Average carelessness has a high impact on accurate STEM predictions. This predictor importance is in line with previous studies that proved the importance of carelessness in case of students opting STEM fields [6,10]. One limitation of this study is related to the use of clickstream data which depends on multiple factors like time spent on the system, the number of questions

answered which may vary when considering different sets of students that work on the platform during different periods. Another limitation is the generalizability of this study as the dataset analyzed is from a single platform (ASSISTments), and the predictor relevance are model specific.

In our future work, we focus on developing feature selection techniques based on the useful predictors and develop models that efficiently and effectively predict their choice based on their middle school year data.

6. REFERENCES

- [1] Bolkan, S., & Griffin, D. J. (2017). Students' use of cell phones in class for off-task behaviors: The indirect impact of instructors' teaching behaviors through boredom and students' attitudes. *Communication Education*, 66(3), 313-329.
- [2] Hall, M. A. (2000). Correlation-based feature selection of discrete and numeric class machine learning.
- [3] Mierswa, I., & Klinkenberg, R. (2019). RapidMiner Studio (9.2) [Data science, machine learning, predictive analytics]. Retrieved from <https://rapidminer.com/>
- [4] Mierswa, I., & Wurst, M. (2006, July). Information preserving multi-objective feature selection for unsupervised learning. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation* (pp. 1545-1552). ACM.
- [5] Nugent, G., Barker, B., Welch, G., Grandgenett, N., Wu, C., & Nelson, C. (2015). A model of factors contributing to STEM learning and career orientation. *International Journal of Science Education*, 37(7), 1067-1088.
- [6] Ocumpaugh, J., San Pedro, M. O., Lai, H. Y., Baker, R. S., & Borgen, F. (2016). Middle school engagement with mathematics software and later interest and self-efficacy for STEM careers. *Journal of Science Education and Technology*, 25(6), 877-887.
- [7] Pedro, M. O., Baker, R., Bowers, A., & Heffernan, N. (2013, July). Predicting college enrollment from student interaction with an intelligent tutoring system in middle school. In *Educational Data Mining 2013*.
- [8] Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N. T., Koedinger, K. R., Junker, B., ... & Livak, T. (2005). The Assistment project: Blending assessment and assisting. In *Proceedings of the 12th Annual Conference on Artificial Intelligence in Education* (pp. 555-562).
- [9] San Pedro, M. O., Baker, R. S., Heffernan, N. T., & Ocumpaugh, J. L. (2015, March). Exploring college major choice and middle school student behavior, affect and learning: what happens to students who game the system?. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge* (pp. 36-40). ACM.
- [10] San Pedro, M. O., Ocumpaugh, J., Baker, R. S., & Heffernan, N. T. (2014). Predicting STEM and Non-STEM College Major Enrollment from Middle School Interaction with Mathematics Educational Software. In *EDM* (pp. 276-279).
- [11] Yeung, C. K., Lin, Z., Yang, K., & Yeung, D. Y. (2018). Incorporating Features Learned by an Enhanced Deep Knowledge Tracing Model for STEM/Non-STEM Job Prediction. *arXiv preprint arXiv:1806.03256*.

A Methodology for Student Video Interaction Patterns Analysis and Classification

Boniface Mbouzaio
Polytechnique Montréal

Michel C. Desmarais
Polytechnique Montréal

Ian Shrier
McGill University

ABSTRACT

Massive open on-line courses (MOOCs) often rely on video as the prime choice of media content. Given their importance, it is no surprise that many studies that focus on the way students engage with the videos within the MOOC emerged in recent years. A methodology is introduced for the detailed analysis of student video watching patterns and compared with a prevailing approach. The method encodes video interaction sequences in a vector space model that defines distance measures between them. A pattern is defined as a centroid of sequences. We apply the method and conduct a study of how students interact with videos by analyzing interaction patterns across different videos. Within this method, the analysis of the influence of video on patterns is framed as a classification task based on Support Vector Machine (SVM), Boosted Tree (GBM) and nearest neighbour approaches. The results reveal there is a significant difference in the patterns of interaction across videos. It demonstrates the usefulness of the methodology in comparison with a simpler and common approach.

Keywords

MOOC, video watching event traces, analytics, clickstream, video feature, student traces, video event.

1. INTRODUCTION

In contemporary versions of MOOCs, videos play a significant role in the delivery of the course content. Videos often represent the main channel for teaching in MOOCs, and possibly for distance learning environments in general (see for eg.[3, 9]). Video interaction events such as *play*, *pause*, *seek*, and *stop*, characterize how a student listened to a video.

While we find a substantial amount of research to determine factors relating to videos that impact student engagement and learning efficiency, we still find relatively few studies on how students interact with videos themselves, beyond simple aggregations of video watching events. This study adds a contribution to fill this gap by introducing a methodology

to study such patterns based on sequences, and by applying it to uncover differences in interaction patterns across videos. To test this methodology, we investigate whether videos induce student interaction patterns. In other words, could a specific video has its own “signature” in terms of video watching pattern?

2. RELATED WORK

We first review the work that relates to the methods of analyzing student logs of video watching sessions, and in particular the basics of the feature based video analysis.

2.1 Student interaction patterns

Video interactions can be obtained from logged events, and we can distinguish between two approaches of analyzing such logs, namely those that preserve sequential information of interaction sessions, and those that extract features from events and eliminate time, or ordering information.

For example, characterizing video interactions in terms of total watch time, total pause duration, percent completion, video speed, etc., obviates the order or the timing of the corresponding events. The alternative approach is to characterize a video interaction as a sequence of events, or activities [4, 11]. The former characterization, we refer to as *feature-based*, retains numeric features or Boolean factors that can be used to predict variables of interest such as engagement, learning style, etc. In the later characterization, *sequence-based*, interactions are time-ordered sequences between which we can measure distances. Our approach falls into this second category and will be exposed later.

An instance of the feature-based approach that extracts information from video interaction sessions is the highly cited study of Guo et al. [5]. Their study is conducted over a very large sample of MOOCs and student sessions. They found that multiple factors associated with videos can affect student engagement, as measured by the length of watching time and the students’ willingness to complete an assessment activity.

Another interesting finding of Guo et al. [5] is that tutorials are often watched multiple times and for longer periods than lectures. This finding hints that some videos can induce a watching pattern.

In a similar vein, Bhat et al. [2] found that slight changes in how the teacher is depicted in a video can influence engage-

Boniface Mbouzaio, Michel Desmarais and Ian Shrier "A Methodology for Student Video Interaction Patterns Analysis and Classification" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 611 - 614

ment in terms of the proportion of video watched, which is another instance of a feature-based analysis.

Moving on to the sequence-based interaction analysis, the Sinha et al. [11] study is particularly relevant and original. Instead of using raw events such as *pause* and *play*, they also look at pairs and n-tuples of events. They develop an n-gram based encoding of video watching events and use subsequences of events in addition to individual events. They attempt to use levels of n-gram interaction and predict engagement, identify arousal states, and likeliness of dropout. Their approach reaches a good level of prediction accuracy with a Kappa score that ranges between 0.5 and 0.9 for the different predictions. A similar approach has also been applied to problem solving student log data [10].

Other instances of sequence-based interaction analysis have been conducted over data from student usage logs of different learning applications [4, 7, 6]. These approaches use a distance measure, generally the minimum edit distance, or Levenshtein distance, to cluster sessions. The sequences of events are transformed into sessions of activities represented by vectors. Each unit of the vector represents a time duration and corresponds to an activity. The clustering of such vectors yields a synthetic view of the different types of sessions.

The current approach is inspired from the sequence-based interaction approaches and uses a distance measure between watching sessions. However, the similarity among sessions is not based on the Levenshtein distance, or edit-distance. Instead, it defines a vector space model. Furthermore, a class label is represented by a point in that space corresponding to the prototypical sequence of the class label. We introduce later the means by which a prototypical sequence of interactions can be defined and demonstrate how it can identify the watching pattern of each specific video. Before, we describe the basics of the feature based video encoding.

2.2 Feature based video encoding

Our approach is compared to the feature based encoding approach. As mentioned, this is a common approach to detect unique patterns and predict factors of interest such as student engagement level, in-video dropout, and course completion or dropout [11]. Below is a summary of the six video watching features we track.

1. **Video length:** Student interaction can be affected by the length of the video. Some studies have shown that shorter videos are much more engaging [5], thus one of the characteristics of student video interaction is the length of the video.
2. **Length of time that student played the video:** This is the total amount of time that the student play a video, including if the student plays the same section of the video more than once. In general, this feature is expressed as a percentage of video play time compared to the video length.
3. **Number of times that the student paused and where the student paused in the video:** Frequent or long pauses may reveal aspects of a video, such as

its level of difficulty, as has been found in some studies [8].

4. **Number of times that the student seeks the video backward:** The backward seeks for a student show how replaying some part of a video is important or the importance of those video sections in the process of student learning.
5. **Number of times that the student seeks video forward:** The forward seek indicates the level of disengagement of the student by skipping sections of the video. For Li et al. [8] either infrequent watching or skipping large sections of the video suggested that the video was perceived to be of a higher level of difficulty.
6. **Number of times the student stopped the video:** In general, the stop event occurs at the end of the video though the student can stop the video at any time.

3. METHODOLOGY TO ENCODE AND CLASSIFY VIDEO INTERACTIONS

In this section, we introduce the method used to encode and classify video interactions that allows a comparison of encoded sequences in terms of distance. Akin to other studies [1, 11, 4], this distance-based approach lends itself to clustering of sequences, but our aim differs from these studies in that we wish to use a labelled data approach. In this study, we will use the videos as the target labels of interest to demonstrate the methodology. The proposed encoding allows a characterization of a video based on a set of instances of watching interaction sequences.

3.1 Video interaction encoding

The general principle is to characterize the video by a prototypical representation, which we will name the *video centroid*, and by which a distance from a student's video interaction sequence can be computed. This will allow us to compute a distance from a student interaction sequence to each of the different video.

In order to achieve such representation, we expand the original sequence of single events by using a vector representation of events. An event is defined as a vector of 5 types of events. For example, a sequence of four events, $s = \{(\text{play})_1, (\text{play})_2, (\text{pause})_3, (\text{stop})_4\}$ out of a set of six event types that also includes *seek forward/backward* events, is represented as:

$$s = [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4] = \begin{matrix} & \begin{matrix} play & play & pause & stop \end{matrix} \\ \begin{matrix} play \\ pause \\ seek\ backward \\ seek\ forward \\ stop \end{matrix} & \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{matrix}$$

Now, assuming we have a set of n sequences, \mathbf{s} , where each sequence is composed of m events ($|\mathbf{s}| = n$ and $|s| = m$), we define the centroid of this set as $\bar{\mathbf{s}} = [\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2, \dots, \bar{\mathbf{e}}_m]$ where $\bar{\mathbf{e}}_i$ is defined as

$$\bar{\mathbf{e}}_i = \begin{bmatrix} \sum_{j=1}^n e_{i,1,j}/n \\ \vdots \\ \sum_{j=1}^n e_{i,5,j}/n \end{bmatrix}$$

Figure 1: Representation student i 's interaction with a video of length of n seconds

1	2	3	4	5	6	7	8	9	...	n	video length
Pl	Pl	Pl	Pl	$Pa.5$	
.	.	.	.	Pl	Pl	Pl	Pl	$Se.-6$.	.	
.	.	.	Pl	Pl	Pl	$Se.3$	
.	Pl	Pl

1	1	1	2	2	2	1	1	0	...	1	play
0	0	0	0	1	0	0	0	0	...	0	pause
0	0	0	0	0	0	0	0	1	...	0	seek-backward
0	0	0	0	0	0	1	0	0	...	0	seek-forward
0	0	0	0	0	0	0	0	0	...	0	stop

A more complex example of a student i sequence of interaction with a video of length n seconds can be represented as in figure 1. In this example, the student uses *seek* actions to navigate, and therefore some parts of the sequence contain more than a single *play* event at a given time.

3.2 Centroid based classification

The centroid of a set of sequences is defined as a prototypical sequence and represents a class. In the experiment below, this set will be the sequences of interactions of a video. Each event of that class sequence contains averages of the 5 types of events. This allows to define the distance between an individual sequence, s , and a prototypical sequence, \bar{s} , as the Euclidean (Frobenius) norm of the difference between the two matrices: $\|s - \bar{s}\|$, since both s and \bar{s} can be considered as matrices.

To determine the extent to which the prototypical sequence of each video is representative and discriminating, we can use SVM, Boosted Tree and a nearest-neighbour approach to classify sequences on the four same length videos, as explained below.

4. EXPERIMENTS

We apply the methodology described in the last section on the question of whether videos induce different patterns of interaction. The question is framed as a classification task: given a set of interaction sequences of a video, determine the corresponding video.

Our study uses the traces of an online course of McGill University in Montreal in the edX platform that had a substantial number of registered students (Table 1). The course Body101x was given by Professor Ian Shrier in the autumn 2015 and winter 2016. We have access to the traces of both sessions.

Session	autumn 2015
Nb. Students	30,640
Nb. videos	138 + 1 live
Nb. Students honor	10,424
Nb. Students passed	970

Table 1: Course and video information

While the whole course contains 139 videos, we retain 4 sets of 4 videos of them for this study, based on the criteria that in each set the duration of videos are the same (within 8 to 10 minutes). The choice of using a subset of same duration is meant to avoid a bias due to duration: since video duration can induce specific patterns and since duration can vary across the videos, selection of same duration videos avoids this potential bias.

For the purpose of comparing the proposed sequence-based approach with the simpler feature-based approach, we replicate the classification task using feature-based approach (section 3.2). The features are represented as a vector containing the mean proportion of each event: For each video, the events proportions of *pause*, *play*, etc., are computed. Classification of a given interaction sequence into one of the 4 videos is done using SVM, Boosted Tree and a nearest-neighbours approaches, akin to the sequence approach.

5. RESULTS

The results of the classification are shown in table 2. The methodology we used is compared with the features-based approach described above. The results show that the sequence-based approach is substantially better than the feature-based for classifying videos. They suggest that the sequence-based method of analyzing video traces yields more precise representations of interaction patterns than the feature-based, at least when applied to the task of characterizing interactions with different videos. This is demonstrated by a classification task, where video interactions are encoded in a vector space model and a given video is classified by SVM, Boosted Tree and a nearest-neighbour approaches, where centroids represent the classes, namely the videos.

6. CONCLUSION

The results show that the sequence-based method has the advantage of providing a more precise encoding of a set of individual student interactions into a "pattern" than the simpler aggregation of feature variables that is often used in studying student interaction traces with videos, or student traces in general. It allows the definition of labeled clusters, the video styles in our case, that contrasts with standard unlabeled clustering which requires the often difficult interpretation of the clusters.

7. LIMITATIONS

The conclusion that videos induce different interaction patterns among students is limited to a single course. While the number of videos in this course is substantial and the MOOC attracted a large number of students over two years, we cannot rule out that other factors than videos induced the difference in interaction patterns. However, we did rule out the possibility that this is linked to a presenter biased, by

Classifier: Approach:	SVM		Boosted Tree		Nearest Neighbour	
	Sequence	Feature	Sequence	Feature	Sequence	Feature
Accuracy	0.45	0.30	0.43	0.41	0.39	0.25
Balanced Acc.	0.61	0.51	0.62	0.60	0.57	0.50
Precision	0.55	0.34	0.46	0.48	0.44	0.06
Recall	0.43	0.21	0.64	0.40	0.36	0.25
F_1	0.44	0.27	0.45	0.42	0.34	0.10
Kappa	0.23	0.02	0.23	0.18	0.13	-0.02

Table 2: Accuracy of tenfold cross validation runs of four videos of same length using feature-based approach and the proposed sequence-based method.

running the classification task over presenters, and the finding that the predictions are substantially lower than for the videos. Nevertheless, we did not go over all other possible covariates that could induce a bias.

To a lesser extent, the demonstration that the sequence-based method of characterizing interactions is more precise than the feature-based is also limited to the sample of this study. However, it is harder to find a reasonable explanation that a covariate is responsible for the higher accuracy of the sequence-based approach. We therefore consider this finding more reliable, albeit still requiring further investigation to bring greater confidence in the findings.

8. REFERENCES

- [1] Yoav Bergner, Deirdre Kerr, and David E Pritchard. Methodological challenges in the analysis of mooc data for exploring the relationship between discussion forum views and learning outcomes. *International Educational Data Mining Society*, 2015.
- [2] Suma Bhat, Phakpoom Chinprutthiwong, and Michelle Perry. Seeing the instructor in two video styles: Preferences and patterns. *International Educational Data Mining Society*, 2015.
- [3] Lori Breslow, David E Pritchard, Jennifer DeBoer, Glenda S Stump, Andrew D Ho, and Daniel T Seaton. Studying learning in the worldwide classroom: Research into edx’s first mooc. *Research & Practice in Assessment*, 8, 2013.
- [4] Michel Desmarais and François Lemieux. Clustering and visualizing study state sequences. In *Educational Data Mining 2013*, 2013.
- [5] Philip J Guo, Juho Kim, and Rob Rubin. How video production affects student engagement: An empirical study of mooc videos. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 41–50. ACM, 2014.
- [6] Jiangang Hao, Zhan Shu, and Alina von Davier. Analyzing process data from game/scenario-based tasks: An edit distance approach. *JEDM-Journal of Educational Data Mining*, 7(1):33–50, 2015.
- [7] Severin Klingler, Tanja Käser, Barbara Solenthaler, and Markus H Gross. Temporally coherent clustering of student data. In *EDM*, pages 102–109, 2016.
- [8] Nan Li, Lukasz Kidzinski, Patrick Jermann, and Pierre Dillenbourg. How do in-video interactions reflect perceived video difficulty? In *Proceedings of the European MOOCs Stakeholder Summit 2015*, number EPFL-CONF-207968, pages 112–121. PAU Education, 2015.
- [9] Daniel T Seaton, Yoav Bergner, Isaac Chuang, Piotr Mitros, and David E Pritchard. Who does what in a massive open online course? *Communications of the ACM*, 57(4):58–65, 2014.
- [10] Tanmay Sinha and Vincent Aleven. Pace, problem solving and progress: Mining interaction pathways in intelligent tutoring systems.
- [11] Tanmay Sinha, Patrick Jermann, Nan Li, and Pierre Dillenbourg. Your click decides your fate: Inferring information processing and attrition behavior from mooc video clickstream interactions. *arXiv preprint arXiv:1407.7131*, 2014.

Towards discovering problem similarity through deep learning: combining problem features and user behavior.

Dominic Mussack
University of Luxembourg
dominic.mussack@uni.lu

Paul Schrater
University of Minnesota
schrater@umn.edu

Rory Flemming
University of Minnesota
flemm053@umn.edu

Pedro Cardoso-Leite
University of Luxembourg
pedro.cardosoleite@uni.lu

ABSTRACT

Automated learning systems allow educators to scale up their efficacy, while personalized systems retain the ability to customize to the individual student. A core issue in developing such adaptive learning systems is to understand how different items (e.g., math exercises) relate to one another, and to exploit this understanding to predict performance on an item. Data-driven approaches aim to discover latent concepts through embeddings that predict similarity between items, typically using only performance data or item data, but not both. While these embeddings are meant to uncover latent concepts (e.g., associativity in mathematics or chemistry), they are better construed as representing topics that reflect the similarity structure in performance or item features. One major difficulty is that embedded concepts may differ only in presentation and not in substance. For example, when learning about numbers, young children struggle with different representational formats (e.g., finger counts, Hindu-Arabic numeral) despite the underlying concept being the same (e.g., “3”). By incorporating item information that allows structured similarity comparison between an item’s content and representational format, we can begin to parse out what aspects lead to behavioral differences. Here we develop a deep learning framework for learning concept embeddings that integrates behavioral and item-features to better factorize embeddings into content and presentation. This allows us to fully represent the complexity of the items space, while still extracting scientifically-useful results from the analysis.

Keywords

education testing, item similarity, deep learning, concept discovery

1. INTRODUCTION

Personalized learning systems use student performance to assess their current knowledge in order to present appropri-

ate questions; understanding item similarity helps in this inference process [6]. When the goal is to predict performance, this similarity measure should most likely correspond to the abilities or skills the items are meant to test (e.g., whether a student knows multiplication or not). Data-driven discovery methods for similarity are needed for larger sets of items [1]. There are many ways to measure similarity [6], however all require that items be represented in the same feature space. This means that one of the most important decisions in computing similarity is the choice of input data, because it determines the meaning of the discovered similarity [7]. Having a rich input feature space which correctly corresponds with the intended application maximizes discovery of relevant relationships. For instance, if a user can struggle with either an item’s content or presentation, then features relevant for both are necessary to determine which impacts performance.

Importantly, raw features can be projected into a latent or embedded representational space. This is the common approach for topic modeling in text analysis [3], which finds latent ‘topic’ representations for documents and words. Typically these embeddings are discovered using either performance or item data, but not both. Using both types of data appropriately is challenging because the method of discovering the embeddings must respect the structural relationship between item features and performance data, and the structure in the item features themselves. To understand the similarity in educational concepts, we want to represent item similarity in terms of how the item features interact with performance data. This is because educational concepts refer to not just intrinsic aspects of the item, but the way that different learners interact with and represent them.

An example that illustrates the difficulty of discovering such embeddings in education is in number representations. Adults can easily switch between various representational formats of numbers (e.g., Hindu-Arabic numerals to mathematical concepts, finger counts to roman numerals, etc) to make assessments [8]. By contrast, young children struggle with some representational formats, which can greatly impact their ability to perform on an otherwise simple problem. This is similar to creating representations that respect the ‘style vs. content’ distinction [9], in that the latent representation must correspond to the structure that is hypothesized in the data.

Dominic Mussack, Rory Flemming, Paul Schrater and Pedro Cardoso-Leite "Discovering item similarity through deep learning: combining item features and user behavior." In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 615 - 618

The difficulties in estimating similarity are partly due to the complexity of item features. We demonstrate this using Mathemarmite (mathemarmite.lu), an educational game designed to teach children numeracy. Mathemarmite requires children to appropriately count using different representations (e.g., digits, fingers) (see figure 1). Each item requires selecting the right number of each ingredient to mix together, with the ingredient count represented in various ways. Trials are structured hierarchically, as a single “recipe” can include multiple representations at once with varying count. In other words, each item can have multiple “sub-items” that must be completed in any order for successful item completion. This produces an added complexity where any method we develop must deal with both variable number of features and permutation invariance [10]; the order of the items completed is arbitrary.



Figure 1: A standard trial in the educational game Mathemarmite. Children change the visuals of the monster (on the right) by correctly following recipes on the scroll. Ingredients from the shelf can be placed in the cauldron, with correct number of each ingredient shown on the recipe using various numeric representations (here tick marks and a die). Item features include the number of lines, line representation, and number of ingredients per line. Mathemarmite uses an adaptive difficulty system, to keep users engaged at their skill level.

Here we develop a deep learning approach towards discovering similarity in Mathemarmite. Our approach is similar to partial least squares [2], in that we find a directed latent space representation where item similarity is constrained based on similar performance. It is also similar to work by [11], who develop a dynamic key-value memory network (DKVMN) which learns a concept embedding for items, and models an evolving knowledge-state of a learner. A similar memory-augmented neural network approach in educational systems was used by [4]. These approaches use long short-term memory (LSTM) networks to compress a user’s history, along with attention-based mechanisms to compute similarity among items. We limit ourselves to a non-dynamic system, given the data size we are working with. Our approach focuses on user-independent concept space (see figure 2), attempting to find a similarity score across users.

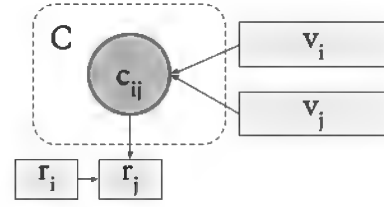


Figure 2: Item features v_j and v_i are used to predict performance on item j by embedding them in a similarity space C . The similarity score c_{ij} is used along with performance on problem i to predict problem j . This constrains the similarity space to finding scores that are similar in terms of their impact on performance.

2. METHODS

2.1 Datasets

We present results using one simulated dataset and one collected educational game dataset (i.e., Mathemarmite). The Mathemarmite dataset is based on 4961 trials from 140 users (after cleaning to remove users with less than 5 trials and users who are not in the target demographic i.e., remove adults). The simulated dataset is constructed to mimic the structure of the Mathemarmite dataset, based on the assumptions of the similarity network (e.g., figure 2). That is, we generate 5 latent “concepts”, represented as 5 Gaussian spheres in concept space (each 5d), and sample item concepts from them. Items are then transformed to produce observed features using simple random linear projection matrices, and then these latent concepts determine performance based on a cosine similarity between the concept latent space and the item’s concept space score, where the latent spaces are weighted based on user performance. If a user is good at a concept, and there is high overlap between that concept and the item’s latent score, then the user has a high performance. We generate 5000 samples from this dataset for 5 users each (25000 items) for training. We also compare training performance on one user versus on all users, referring to them as “Multi user simulated data” and “Single user simulated data”.

2.2 Targeted similarity network

2.2.1 Network Architecture

Here we develop a deep learning framework for learning concept embeddings that integrates behavioral and item-features (see figure 3). We construct a targeted similarity network that learns an embedded representation for similarity comparison, based on item performance. We attempt to learn a similarity space for item features v_i by learning a predictive model $p(r_i|v_i, v_j, r_j) = \sigma(c_{ij} \times r_j)$, that is we want to predict the performance of problem i from problem j and the similarity between the two (where σ is a sigmoid function).

Categorical features are embedded into a linear space, using a learned embedding matrix E_1 . We then employ a deep set architecture to deal with permutation invariance in the subproblems. Consider an item that involves a series of subproblems to be solved, each of which can be solved in any

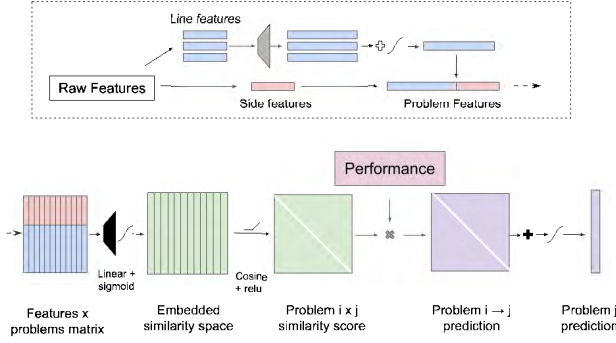


Figure 3: Top: Network feature embedding component, where embedded features are then passed into the main network. Features are processed through these steps, separating line features to deal with permutation invariance, and then concatenated with side features. Bottom: Main network embedding, where features are embedded into a similarity space and similarity scores between all items are computed for prediction.

order. We can then consider the problem P to be composed of subproblems p_k , and side features s . Therefore: $P_i = \{p_1, p_2, \dots, p_m, s\}$, where the indices k of the subproblems may be swapped, and m is variable across problems (i.e., the number of sub-problems can vary, such that P_i might include only a single subproblem). Each subproblem has an associated feature vector of length l . As mentioned above, each subproblem has one categorical feature labeled a_k , we embed with the same matrix (note $p_k = [a_k, z_k]$ with non-categorical z_k features). To model the permutation invariance, we use sum-decomposition via latent space as in [10], to produce a single vector per problem that represents the combined influence of all subproblems.

Each problem also has a vector of side features, which includes a feature coding the number of subproblems. The decomposed subproblems are then concatenated with the side features to produce an overall item feature set, which we label x_i for problem i . We can describe this initial feature embedding process with the set of functions (see figure 3):

$$\hat{a}_k = E_1^T a_k \text{ for each } a_k \text{ in } P_i \quad (1)$$

$$v_i = \sigma(\text{sum}_m(E_2^T \hat{P}_i)) \quad (2)$$

$$x_i = [v_i, s_i] \quad (3)$$

where \hat{a}_k are the embedded categorical features for each subproblem and E_2 is the linear weights for the sum-decomposition (size l by e_2 latent space). The sum is down the same dimension as the stacked feature vectors, such that v_i is length e_2 . Then we compute item similarity by projecting the embedded feature matrix X to the similarity space S , and take the cosine similarity of S with itself: Item performance is then predicted off of the similarity score and other item's performance (represented -1 or 1 for incorrect, correct), and passed through a sigmoid function. Items with high similarity are then predicted to have similar performance, while those with low similarity are not. Since our interest is in

Table 1: Final AUC Test Scores

Model	Dataset	AUC Score
Network	Single-user sim	~ 0.999
Network	Multi-user sim	0.998
Network	Mathemarmite	0.518
Subject average	Mathemarmite	0.762
Item average	Mathemarmite	0.541

the similarity space itself, we allow all items in a batch to predict the performance of all other items, by adding their prediction.

$$S = \sigma(W^T X) \quad (4)$$

$$C = \text{sim}(S) \quad (5)$$

$$\hat{R} = CR \quad (6)$$

Where R is a batch-size square matrix of the response vector copied down the rows. $\text{sim}(S)$ computes the similarity score of all items using positive cosine similarity, $\text{sim}(s) = \text{relu}(\hat{S}\hat{S}^T)$ where $\hat{S} = \frac{S}{\|S\|}$. This allows C to take on the interpretation of item by item similarity matrix where C_{ji} (jth row and ith column) being the $i \rightarrow j$ similarity for prediction. The final prediction is then made by summing across \hat{R} column space (ignoring the diagonal $i \rightarrow i$ similarity), that is $\sum_i \hat{R}_{ji}$ to produce a vector of predicted j responses when passed through a final sigmoid output.

3. RESULTS

All network experiments are coded using Pytorch. Networks are trained using Binary Cross Entropy loss. Hyperparameter selection is done via cross-validated AUC scores on 100 runs, finding an embedded similarity space of 7 for Mathemarmite and 5 for both simulated datasets. We then train these networks on 500 runs (epochs) with those parameters using a 20% train-validation split (split trials selected at random), showing both train and validation AUC scores in figure 4.

We also compared against two baseline models: predicting performance off of each subject's average performance, and predicting performance off of each items's average performance (leaving out the given trial in both cases). Final AUC test scores are shown in table 1.

Note that AUC of 0.5 is by chance with 1 being perfect, so the Mathemarmite dataset is just above chance performance.

3.1 Visualization

The significant value of this network is in being able to construct a similarity space to compare the items, and then interpret the resulting space. To visualize the space, we use t-SNE [5], which allows us to project the similarity weights of items down to a 2D space for visualizing clusters. As shown in figure 5, we are able to reconstruct our latent concept space in the simulated dataset using the network (single-user). Similar projection exists for multi-user dataset.

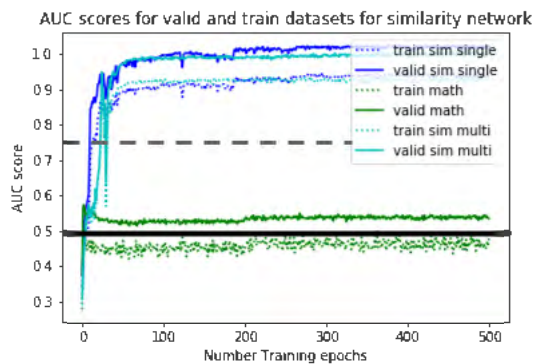


Figure 4: AUC scores across training epochs. Black line indicates chance performance, while dashed line is baseline prediction in the Mathemarmite dataset (average subject performance).

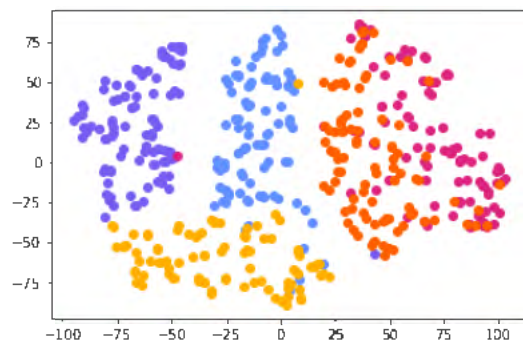


Figure 5: Recovered latent spaces in simulated data (single user). Colors correspond to arbitrary latent concept.

While we do not have ground truth for the Mathemarmite dataset, we can still perform the same visualization. However, given our final performance score, we cannot directly interpret the resulting space, and it is therefore not a meaningful latent concept space.

4. CONCLUSION

In this paper we developed a novel network for learning a similarity space for educational test data, and applied it to simulated and real data. Given that the performance on the simulated dataset is high, the network is able to reconstruct the latent space of items appropriately, given the assumptions of the underlying data generative process. However, the network does not perform well on the collected dataset. Given that we can predict line performance using a standard logistic regression model (AUC at ~ 0.65 , where line features predict line performance), item features are informative of line performance, and therefore item performance (recall that items are made of multiple lines). A possible issue with the Mathemarmite dataset is that we intermix different users, with high variability across users. Given the adaptive algorithm that underlies Mathemarmite, only players with higher performances will attempt otherwise harder problems, making it more difficult to separate user and item relations.

An important limitation of our current model is it does not incorporate individual user differences in terms of the found similarity space, therefore it cannot account for changes in a user performance as well (i.e., due to learning); our network cannot learn what it cannot represent. An alternative is to incorporate user-level features, and structure the network in a bilinear fashion, much like standard item response theory. In other words, expert users likely have a different concept space than novice users. Future work involves developing a network that instead allows multiple similarity spaces across users. Such additional complexity should capture the underlying structure of the Mathemarmite dataset and lead to insights on number representational space.

5. ACKNOWLEDGEMENTS

This research was supported by the Luxembourg National Research Fund: ATTRACT/2016/ID/11242114/DIGILEARN and INTER Mobility/2017-2/ID/11765868/ULALA grants

6. REFERENCES

- [1] Behdad Bakhshinatogh, Osmar R Zaiane, Samira ElAtia, and Donald Ipperciel. Educational data mining applications and tasks: A survey of the last 10 years. *Education and Information Technologies*, 23(1):537–553, 2018.
- [2] Paul Geladi and Bruce R Kowalski. Partial least-squares regression: a tutorial. *Analytica chimica acta*, 185:1–17, 1986.
- [3] Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.
- [4] Qi Liu, Zai Huang, Zhenya Huang, Chuanren Liu, Enhong Chen, Yu Su, and Guoping Hu. Finding similar exercises in online education systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1821–1830. ACM, 2018.
- [5] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.
- [6] Radek Pelanek. Measuring similarity of educational items: An overview. *IEEE Transactions on Learning Technologies*.
- [7] Jirí Rihák and Radek Peláněk. Measuring similarity of educational items using data on learners’ performance. In *EDM*, 2017.
- [8] Roger N Shepard, Dan W Kilpatrick, and James P Cunningham. The internal representation of numbers. *Cognitive psychology*, 7(1):82–138, 1975.
- [9] Joshua B Tenenbaum and William T Freeman. Separating style and content with bilinear models. *Neural computation*, 12(6):1247–1283, 2000.
- [10] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pages 3391–3401, 2017.
- [11] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, pages 765–774, 2017.

Machine Learning Based Decision Support System for Categorizing MOOC Discussion Forum Posts

Gaurav Nanda
gnanda@purdue.edu

Kerrie A. Douglas
douglask@purdue.edu
School of Engineering Education
Purdue University, West Lafayette, US

ABSTRACT

In this paper, we propose and evaluate a decision support system (DSS) to predict the topic of discussion forum posts among categories: *Social*, *Concept*, *Assignment*, and *Others*, in Massive Open Online Courses (MOOCs). The DSS uses two supervised machine learning (ML) models- Logistic Regression and Multinomial Naïve Bayes to output: two most-likely categories with associated prediction probabilities, and a surety indicator (High, Medium, or Low) of prediction. We evaluated the DSS on manually-labelled discussion posts from two programming related MOOCs offered on different platforms and the ML models being trained on the same and different course. Our results indicate that: a) the prediction performance of DSS is better when trained on labelled data from the same course, b) when trained on data from another course, the DSS performs better if training data is sampled to reduce the bias in distribution of categories, and c) providing the two most-likely categories is helpful for indicating the smaller categories that are usually difficult to predict by ML models. This DSS can help instructional teams to better manage the MOOC discussion forum and learners to easily navigate and find relevant posts.

Keywords

MOOC Discussion Forum, Text Classification, Machine Learning, Decision Support System, Logistic Regression, Naïve Bayes

1. INTRODUCTION

Discussion forums are the primary medium of social interaction in MOOCs and have been found to help improve learners' understanding of course material as well as provide a community learning experience in an online environment [1]. Involvement of instructor and teaching assistants can encourage learner participation and improve quality of discussion [2]. However, managing the discussion forum of a large MOOC can be challenging for the instructional team due to large number of participants and multiple ongoing threads. Similarly, for learners, it can be difficult to navigate through and find relevant posts. A DSS that can organize the discussion forum posts topic-wise and highlight posts that require attention can be very helpful.

Some MOOC platforms, such as EdX, ask users to input the topic of discussion by selecting from a predefined list of topics when

creating a discussion post [4], but this approach relies on users to correctly input the topic and having an exhaustive list of predefined topics ready. Earlier studies have examined various approaches for addressing the issue of unorganized MOOC discussion forums [3], such as: (a) classifying content related discussion forum posts using machine learning for models trained on same course and different course [5]–[8] (b) identifying user posts that need instructor attention using supervised machine learning models [1] [9], (c) analyzing discussion posts using topic modelling [10], and (d) identifying question related threads and their potential answers in discussion forum [11].

Previous studies have used different levels of categorization of MOOC discussion posts, such as, “content-related” and “non-content related” [7], relatively broader categories such as “Course material”, “Course Logistics”, and “General (all other types)” [10], or more specific set of categories such as “Content”, “Other Coursework”, “Social/affective”, “Technology”, “Policies” and “Other” [12]. There are various advantages and challenges associated with very broad or very specific categories of discussion posts in terms of informativeness and prediction capability of machine learning (ML) models. While very broad categories can be easily predicted by ML models, there can be multiple sub-categories within each of the broad category. On the other hand, very specific categories are more informative, but predicting them accurately using ML can be challenging due to issues such as imbalanced data distribution among categories and very few training cases of smaller categories. Depending on the purpose of classification and availability of training data, an optimum set of categories should be determined.

2. PROPOSED SYSTEM

The DSS combines the prediction outputs of two classical ML models- Multinomial Naive Bayes (MNB) [13] and Logistic Regression (LR) [14] to predict the category of discussion based on the first post of a discussion thread, as shown in Figure 1.

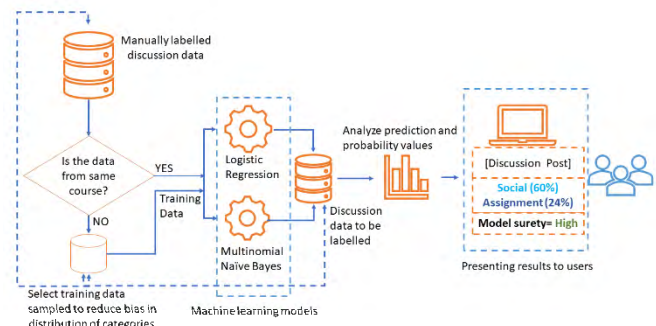


Figure 1: Schematic diagram of proposed DSS

We selected these two ML models as they both have been widely used for text classification, are scalable, and output the probability

Gaurav Nanda and Kerrie Douglas "Machine Learning Based Decision Support System for Categorizing MOOC Discussion Forum Posts" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 619 - 622

of prediction along with the predicted category- which is an important component of the proposed DSS, that has been similarly used in healthcare and other domains [15], [16], [17]. The prediction probability is informative to the end-user as it indicates the confidence in predictions. The DSS prediction probability- $Prob_DSS(Category)$ for each category is derived by averaging its prediction probabilities outputted by the LR and MNB models. The DSS outputs two most-likely categories along with their corresponding $Prob_DSS(Category)$ based on largest and second largest $Prob_DSS(Category)$ values.

The DSS also outputs a model surety indicator- DSS_Model_Surety that can assume values *High*, *Medium*, and *Low* based on following conditions: a) if predictions of LR and MNB do not agree, then *Low*, b) if MNB and LR predictions agree and $Prob_DSS_Max$ (i.e., maximum $Prob_DSS(Category)$ among all categories) < 0.5 , then *Medium*, and c) if MNB and LR predictions agree and $Prob_DSS_Max \geq 0.5$, then *High*. The threshold for $Prob_DSS_Max$ was set to 0.5 as it meant that both LR and MNB predicted that category with relatively high confidence. This threshold can be changed depending on the number of categories and objective of DSS. As shown in Figure 1, the DSS uses a strategy for providing training data to the ML models depending on availability of manually-labelled data from same course. This process is also discussed later in the paper.

3. METHODS

We used manually annotated discussion forum posts from two computer programming related MOOCs offered on different platforms, both with approximately 4800 enrollees, referred to as FL and EdX in the rest of the paper. FL, titled “R for Data Science”, was offered on the FutureLearn platform in Spring 2017 and had about 900 discussion threads with multiple posts in each thread. EdX, titled “JAVA Programming”, was offered on EdX in Summer 2017 and had about 300 discussion threads.

For assigning categories to each thread, three researchers independently coded a sample of 100 threads and then compared and refined the emergent themes. Using the refined themes, they independently assigned codes to a different sample of 50 comment threads and reconvened to finalize the themes that describe the main content of threads. These themes were: *Social* (introductions, reasons to take the course, expectations, experience etc.), *Concept* (questions and comments about course content covered in lecture videos or readings), *Assignment* (discussion related to assessments), and *Other* (course policy, website issues, other technical issues, general feedback, and anything that did not fit in remaining categories). Using this coding scheme, all discussion threads for both courses were analyzed such that each thread was assigned a code by two researchers. Any disagreement was reviewed by all three researchers and a consensus was achieved.

We used Sensitivity and Positive Predictive Value (PPV) as the prediction performance measures for individual categories:

$$Sensitivity = \text{True Positives} / (\text{True Positives} + \text{False Negatives})$$

$$PPV = \text{True Positives} / (\text{True Positives} + \text{False Positives})$$

For evaluating the overall performance, we used the measures WAvG-Sen (the overall Sensitivity, which is weighted by the size of the category), UAvG-Sen (simple average of Sensitivity across all categories, that does not account for category size), and UAvG-PPV (simple average of PPV across all categories). For classification purposes, we used only the first posts of the discussion thread, as done by [5], because the primary topic of discussion is usually set by the first post of the thread, although it

can sometimes change later on. This approach can also help the instructional team in prioritizing their responses to questions. We implemented the MNB and LR models using the machine learning package WEKA [18] and its LIBLINEAR library [19] for L2-Regularized Logistic Regression model. We preprocessed the text by deleting non-alphanumeric characters and removing stopwords using the Rainbow [20] stopword list in WEKA. We conducted following set of experiments for performance evaluation:

Experiment 1: We first evaluated the baseline performance of the ML models when trained on data from same course. We used the 10-fold cross validation method to calculate the Sensitivity and PPV of LR and MNB models for both courses- FL and EdX.

Experiment 2: We then examined the performance of ML models trained on data from a different course on a different platform in a similar area. We used the entire data from one course as the training set and the entire data from another course as the prediction set resulting in following combinations of training and prediction (test) sets: Train: EdX, Test: FL and Train: FL and Test: EdX.

Experiment 3: FL and EdX had significantly different distribution of categories, which can lower the prediction performance of ML models when trained on data from another course. We examined the performance of ML models with modified training sets having similar distribution of categories as prediction set by under-sampling cases from high-frequency categories. We did not aim to match the exact distribution of the prediction set; our idea was to reduce the bias in the distribution of categories at a broader level. The composition of original (FL and EdX) and modified training sets (FL-Mod and EdX-Mod) is shown in Figure 2. The resulting combinations were: Train: EdX-Mod, Test: FL and Train: FL-Mod and Test: EdX.

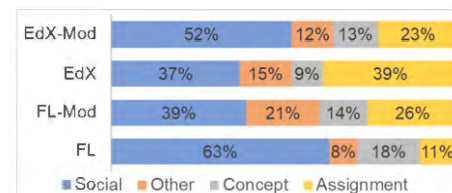


Figure 2: Distribution of categories in training datasets

Experiment 4: Both courses were divided into random non-overlapping training (80% data- EdX-train and FL-train) and prediction sets (remaining 20% data – EdX-test and FL-test). The performance of ML models and DSS were examined on:

- training data from same course, i.e., following combinations- Train: EdX-train, Test: EdX-test, Train: FL-train, Test: FL-test.
- training data from another course sampled to have similar distribution of categories, i.e. following combinations- Train: EdX-Mod, Test: FL-test, Train: FL-Mod, Test: EdX-test.

4. RESULTS and DISCUSSION

Experiment 1: The 10-fold cross validation results of LR and MNB models on FL and EdX are presented in Figure 3.

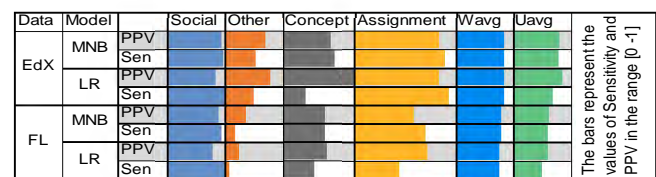


Figure 3: Prediction Results of 10-fold Cross Validation

As shown in Figure 3, both ML models had similar overall prediction performance (columns WAvG and UAvG) for EdX and

FL. Among different categories, performance for the larger categories, i.e., *Social* and *Assignment* for EdX, and *Social* for FL was relatively better than smaller categories. We can see a larger gap between the WAvg (Sen and PPV) and corresponding UAvg for FL, which is probably due to the heavily imbalanced distribution of categories in FL, with 63% cases belonging to a single category- *Social*. It is also to be noted that the prediction performance of *Other* is lowest in both the courses. It may be due to the catch-all nature of definition of the category *Other* i.e. posts that are not classifiable in any other category, which can lead to a mix of different kinds of posts in this category and increases the chances of misclassification.

Experiments 2 and 3 results are presented in Figure 4.

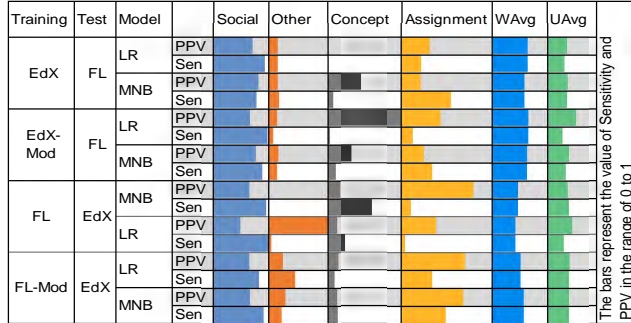


Figure 4: Prediction performance of ML models when trained on data from another course with different sampling methods

As shown in Figure 4, the prediction performance of ML models was considerably better when they were trained on the modified training sets as compared to entire data from another course, most noticeably in case of Test: EdX-test, with Train: FL and Train: FL-mod, possibly because FL data distribution was more skewed.

Experiment 4: First, the performance of ML models for training sets from same and another course are presented in Figure 5.

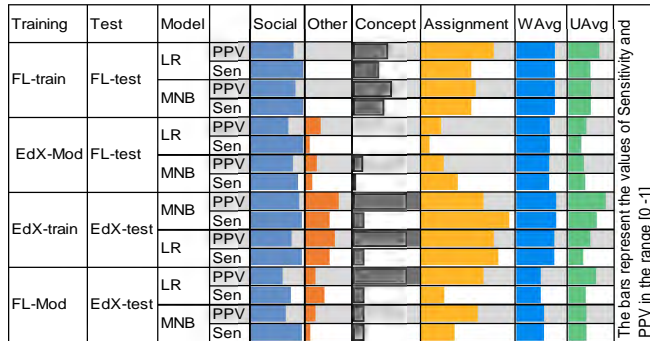


Figure 5: Prediction performance of ML models when trained on same and different course

As shown in Figure 5, both ML models yielded similar overall prediction performance with training data from same course (WAvg-Sen range 0.76-0.79), which was considerably better than training data from another course -- about 10% better WAvg-Sen for Train: EdX-Mod and Test: FL-test, and about 20% better WAvg-Sen in case of Train: FL-Mod and Test: EdX-test. It is to be noted that neither LR nor MNB model always performed better than the other. Therefore, it is advantageous to use predictions made by both ML models in the DSS for robustness-- which was also evident in our results. The prediction performance of DSS' top-prediction is presented in Figure 6. From Figures 5 and 6, we can observe that the DSS performed better than the individual ML models (on both WAvg and UAvg) when trained on data from

another course. For training on data from same course, the DSS performed close to the better-performing ML model.

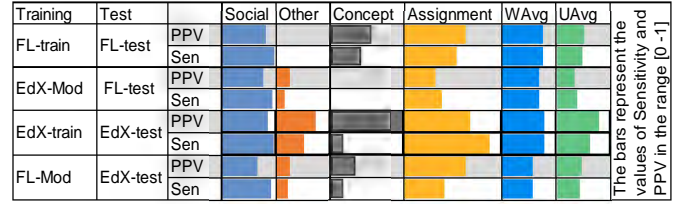


Figure 6: Prediction performance of DSS' top-prediction

The smaller categories- *Other* and *Concept* were most difficult to predict for ML models as well as DSS (top-prediction) even with training data from same course. In order to address this issue, we designed the DSS to provide top-two predictions to users with associated probabilities. The intuition behind this approach was: for the cases corresponding to smaller categories, the second most-likely category would indicate the correct category.

Top-2 Predictions: The above intuition was validated by analyzing the top-two predictions of the DSS as shown in Figure 7.

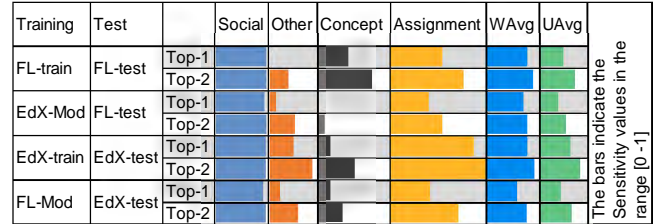


Figure 7: Results for DSS Top-1 and Top-2 predictions

We observed considerable improvement in Sensitivity of the smaller categories (*Other*, *Concept*, and *Assignment*) between Top-1 and Top-2 predictions. Sensitivity for Top-2 predictions were calculated by counting cases in prediction set as 'True-Positive' if the manually assigned category was among DSS' top-2 predictions.

Model Surety: As mentioned before, the Model Surety is determined based on ML model agreement and *DSS_Prob_Max* value. In Table 1, for conditions where the ML models were trained on data from different courses, we present the distribution (Minimum, Maximum, and Average) of *DSS_Prob_Max* values for different scenarios where the LR and MNB models agree or disagree, and if the predictions were correct or not. In Table 1, column N indicates number of cases in test set for each criterion.

Table 1: Range of *DSS_Prob_Max* for different scenarios

Case ID	Predictions Correct?	LR = MNB?	Train: FL-Mod Test: EdX-test				Train: EdX-Mod Test: FL-test			
			DSS_Prob_Max				DSS_Prob_Max			
			N	Min.	Max	Avg.	N	Min.	Max	Avg.
C1	Y	Y	23	0.37	0.94	0.75	98	0.47	0.97	0.75
C2	N	Y	13	0.35	0.58	0.5	32	0.41	0.95	0.63
C3	Y	N	10	0.3	0.61	0.46	13	0.34	0.58	0.42
C4	N	N	12	0.27	0.48	0.39	18	0.32	0.61	0.4

As shown in Table 1, when the model predictions agreed (case ids C1 and C2) the chances of the prediction being correct were higher 64% (23/36 cases) for EdX-test, and 75% (98/130 cases) for FL-test, as compared to when model predictions did not agree- 45% (10/22 cases) for EdX-test and 42% (13/31 cases) for FL-test. It is also to be noted that the average *DSS_Prob_Max* values were considerably higher when the ML models agreed and the predictions were correct, as compared to other scenarios. This means that when the DSS would indicate *High* Model Surety, it is likely to give highly accurate predictions. This is also demonstrated

in the results presented in Table 2, where we show the overall DSS prediction performance (Top-1 and Top-2 predictions) for different Model Surety values when the ML models were trained on data from another course. In Table 2, column N represents number of cases for each criterion, TP indicates True Positives, and %Correct indicates proportion of correctly predicted cases, i.e., TP/N.

Table 2: DSS prediction performance at diff Model Surety

Data		Model Surety	N	Top-1		Top-2	
Train	Test			TP	%Correct	TP	%Correct
EdX-Mod	FL-test	High	122	96	0.79	106	0.87
		Med	8	2	0.25	2	0.25
		Low	31	13	0.42	17	0.55
FL-Mod	EdX-test	High	28	21	0.75	24	0.86
		Med	8	2	0.25	5	0.63
		Low	22	10	0.45	15	0.68

As shown in Table 2, when the Model Surety was *High*, %Correct was substantially higher than when the Model Surety was *Med* or *Low*. It is also to be noted that there were fewer cases (N) with *Med* surety as compared to *Low* surety because when the ML models disagreed, then *DSS_Prob_Max* was likely to be less than 0.5.

5. CONCLUSIONS

In this study, we proposed a machine learning based DSS to predict the broad category of MOOC discussion forum thread based on its first post and provide explanatory output to users. Based on the preliminary evaluation of the DSS, we can conclude that the DSS would perform better when manually annotated data from previous runs of the same MOOC is available. Annotated data from a similar course can also result in reasonable performance if training data sampling is done to remove the bias in data distribution, and, if possible, match the distribution in prediction set. The results also indicated that by providing the top-2 predictions, the DSS can tackle the issue of poor prediction accuracy of smaller categories by ML models. Because of the simple underlying ML models, the DSS is very scalable and can be easily integrated into any existing MOOC platform with suitably customized user interface. The proposed DSS would be helpful in organizing discussion forums better and thus improving learner participation. Future work can examine topic models for estimating the distribution of categories in unlabeled data, including other ML models in the DSS, and conducting user studies on its utility and usability in MOOCs.

6. ACKNOWLEDGMENTS

Our thanks to FutureLearn and EdX for providing us data. This work was made possible by the National Science Foundation (NSF) (PRIME #1544259). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

7. REFERENCES

- [1] O. Almatrafi, A. Johri, and H. Rangwala, "Needle in a haystack: Identifying learner posts that require urgent response in MOOC discussion forums," *Comput. Educ.*, vol. 118, pp. 1–9, Mar. 2018.
- [2] J. Kim, "Influence of group size on students' participation in online discussion forums," *Comput. Educ.*, vol. 62, pp. 123–129, Mar. 2013.
- [3] J. Peral, A. Ferrandez, H. Mora, D. Gil, and E. Kauffmann, "A Review of the Analytics Techniques for an Efficient Management of Online Forums: an Architecture Proposal," *IEEE Access*, pp. 1–1, 2019.

- [4] "How do I add a post in the discussion forum? – edX Help Center." [Online]. Available: <https://support.edx.org/hc/en-us/articles/360002095553-How-do-I-add-a-post-in-the-discussion-forum->. [Accessed: 15-Feb-2019].
- [5] Y. Cui and A. F. Wise, "Identifying Content-Related Threads in MOOC Discussion Forums," in *Proceedings of the Second (2015) ACM Conference on Learning @ Scale - L@S '15*, 2015, pp. 299–303.
- [6] A. F. Wise, Y. Cui, and J. Vytasek, "Bringing order to chaos in MOOC discussion forums with content-related thread identification," in *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge - LAK '16*, 2016, pp. 188–197.
- [7] A. F. Wise, Y. Cui, W. Jin, and J. Vytasek, "Mining for gold: Identifying content-related MOOC discussion threads across domains through linguistic modeling," *Internet High. Educ.*, vol. 32, pp. 11–28, Jan. 2017.
- [8] A. F. Wise and Y. Cui, "Learning communities in the crowd: Characteristics of content related interactions and social relationships in MOOC discussion forums," *Comput. Educ.*, vol. 122, pp. 221–242, Jul. 2018.
- [9] S. Chaturvedi, D. Goldwasser, and H. Daumé III, "Predicting Instructor's Intervention in MOOC forums," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014, pp. 1501–1511.
- [10] A. Ramesh, D. Goldwasser, B. Huang, H. Daume, and L. Getoor, "Understanding MOOC Discussion Forums using Seeded LDA," in *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications*, 2014, pp. 28–33.
- [11] L. Hong and B. D. Davison, "A classification-based approach to question answering in discussion boards," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09*, 2009, p. 171.
- [12] G. S. Stump, J. Deboer, J. Whittinghill, and L. Breslow, "Development of a Framework to Classify MOOC Discussion Forum Posts: Methodology and Challenges," in *NIPS Workshop on Data Driven Education*, 2013, pp. 1–20.
- [13] "A comparison of event models for Naive Bayes text classification," 1998, vol. 752, pp. 41–48.
- [14] *Applied Logistic Regression*. John Wiley & Sons, 2004.
- [15] D. West, P. Mangiameli, R. Rampal, and V. West, "Ensemble strategies for a medical diagnostic decision support system: A breast cancer diagnosis application," *Eur. J. Oper. Res.*, vol. 162, no. 2, pp. 532–551, Apr. 2005.
- [16] X. Liu, R. Lu, J. Ma, L. Chen, and B. Qin, "Privacy-Preserving Patient-Centric Clinical Decision Support System on Naïve Bayesian Classification," *IEEE J. Biomed. Heal. Informatics*, vol. 20, no. 2, pp. 655–668, Mar. 2016.
- [17] G. Nanda, K. M. Grattan, M. T. Chu, L. K. Davis, and M. R. Lehto, "Bayesian decision support for coding occupational injury data," *J. Safety Res.*, vol. 57, pp. 71–82, Jun. 2016.
- [18] "Weka," Springer US, 2005, pp. 1305–1314.
- [19] *LIBLINEAR: A Library for Large Linear Classification*. 2008.
- [20] "Rainbow." [Online]. Available: <http://www.cs.cmu.edu/~mccallum/bow/rainbow/>. [Accessed: 06-Dec-2016].

Hybrid Deep Neural Networks to Predict Socio-Moral Reasoning skills

Ange Tato
Université du Québec à
Montréal
Montréal, Canada
angetato@gmail.com

Roger Nkambou
Université du Québec à
Montréal
Montréal, Canada
nkambou.roger@uqam.ca

Aude Dufresne
Université de Montréal
Montréal, Canada
aude.dufresne@umontreal.ca

ABSTRACT

In this paper we propose a hybrid architecture combining Deep Neural Network architectures with expert's knowledge to automatically evaluate socio-moral reasoning maturity. Socio-moral reasoning represents a key ability to sustain efficient adaptive social interactions. In the proposed solutions, expert knowledge is first computed using NLP and information retrieval techniques and then added to the DNN architecture using the attentional mechanism. It uses pre-annotated textual data and a coding scheme (SoMoral) applied by experts in psychology. State-of-the-art text classification algorithms (Support Vector Machine, Naive Bayes, etc.) achieved very low results in our context in contrast to LSTM and CNN (when combined with expert knowledge). Our solutions were compared to four models: a CNN-only model, an LSTM-only model, a CNN and an LSTM models where the expert knowledge is concatenated directly with the data (instead of using attention). Experiments show that our models can accurately predict the level of socio-moral reasoning skills. Our findings suggest the need for hybrid neural networks that integrate prior expert knowledge (especially when it is necessary to compensate the strong dependency - of deep learning methods - on data size or the possible unbalanced datasets).

Keywords

Convolutional Neural Networks, Long Short Term Memory, Socio-Moral Reasoning skill, Serious Game, Learner Model, Expert Knowledge, Hybrid Neural Networks.

1. INTRODUCTION

Socio-Moral Reasoning (SMR) is a socio-cognitive construct essential for decision-making, as well as social interaction adaptation. It is commonly defined as "how individuals think about moral emotions and conventions that govern social interactions in their everyday lives" [1]. Being able to predict and diagnose one's socio-moral reasoning skill level (or ability) is a key step for quantifying peoples' so-

cial functioning and can be used to identify those at risk for maladaptive social behaviour. This diagnosis could help orient people towards appropriate services or provide adequate support to improve this skill's development. The Socio-Moral Reasoning Aptitude Level (So-Moral)[4] task is a computer-measured walkthrough in which children and adolescents are presented with visual social dilemmas from everyday life. They are then asked to verbalize how they would react in this situation, justifying their answer. The participants' answers are recorded verbatim in transcripts that are subsequently scored manually by experts using a moral-maturity coding scheme inspired by the Kohlberg's theory of moral development [7]. Verbatims are short or long text containing at least one sentence. Each socio-moral reasoning rating was well documented by experts using annotated data. The goal was to put together these two kinds of data to build an accurate model for the automatic prediction of reasoning skill level. Using annotated data only would not give us good results in this context. Therefore we propose a hybrid model that combines expert knowledge with DNN (Deep Neural Networks) architectures (especially CNN and LSTM) using the attentional mechanism [11] for predicting the level of SMR skill of an individual based on his justifications when solving socio-moral dilemmas. The developed solution extends the learner/player model in a serious game called LesDilemmes [13].

2. RELATED WORK

Deep learning architectures are data-driven techniques, therefore their performance is strongly dependent on the amount and the quality of the data. However, many contexts have only a limited amount of data while providing prior expert knowledge. Towel et al. [14] defined hybrid learning techniques as "methods that use theoretical knowledge of a domain and a set of classified examples to develop a method for accurately classifying examples not seen during training". By doing so, a hybrid learning system should learn more effectively than systems that make use of only one of the information sources. There exist very little research focusing on the combination of the a priori expert knowledge and deep learning architectures. Among them, Towel et al. [14] (which might be the first paper to discuss this matter) proposed a hybrid system called KBANN (Knowledge-Based Artificial Neural Networks). It maps expert knowledge, represented in propositional logic, into neural networks and then refines this reformulated knowledge using back-propagation. Coro et al. [3] combined Neural networks (NN) with simulated expert knowledge. Zappone et al. [16]

Ange Adrienne Nyamen Tato, Roger Nkambou and Aude Dufresne "Hybrid Deep Neural Networks to Predict Socio-Moral Reasoning skills" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 623 - 626

recently did the same by combining expert knowledge and Artificial Neural Networks (ANN) in order to optimize wireless communication networks.

In the educational domain, a priori expert knowledge is usually available and generally used to build Intelligent Tutoring Systems (ITS). In other domains, expert knowledge can be available through books or previously built models (such as rules-based models). We believe that this a priori expert knowledge, sometimes acquired over decades of intense research, cannot be dismissed and ignored. In the present paper, in particular, we put forth an approach that uses attentional mechanism and capitalizes on the availability of (possibly simplified or inaccurate) theoretical models in order to reduce the amount of empirical data to use and the complexity of deep artificial training neural networks. To our knowledge no research has proposed to combine a priori knowledge with deep learning architecture using the attentional mechanism as of yet. Moreover, no research in the educational data-mining domain has yet focused on this matter despite the availability of expert knowledge. We applied the proposed solution to the automatic detection of socio-moral reasoning skill level of learners (players) in a serious game.

3. CONTEXT

One of the objectives underlying the development of the proposed model is to implement the automated scoring mechanism in a serious video game called LesDilemmes [13]. It is a first-person serious game which aims to assess and train the social reasoning skills of the player. Through this work, we aim to build an effective model of the socio-moral facet of the learner/player. The level of socio-moral reasoning skill of an individual is determined from its verbal justifications provided when solving the dilemmas. This involves the implementation of a model for automatic measurement of this level during the game. We have a dataset of verbatims coming from the SoMoral experimentation already annotated by experts and a description (a paragraph with key concepts) associated with each different level (or class) of maturity (skill). This paper describes a machine learning model that can accurately assess the socio-moral reasoning skill level of a player based on his verbatim.

The original So-Moral task includes five different levels of socio-moral reasoning skill[1] : (1) Authoritarian-based consequences, (2) Egocentric exchanges, (3) Interpersonal Focus, (4) Societal Regulation and (5) Societal Evaluation. Transition levels (i.e. 1.5, 2.5, 3.5, 4.5) are used to account for verbatims that provide elements of two reasoning stages and show a sequential progression from one stage to another. Occasionally, a verbatim is assigned to two different closed levels (1 being the maximum deviation) when two independent experts annotate the data for rater reliability purposes. A first convolutional neural network has been proposed to automatically detect the socio-moral reasoning skill. Despite the high accuracy reported (92 %) [13], the model was not able to predict when a person has a socio-moral reasoning level of 2, 3 or 5. All the predictions were either levels 1 or 3 due to the unbalanced dataset. This paper addresses this issue in order to ensure the accuracy of the learner model for a better support of the learning process in the serious game.

4. THE PROPOSED HYBRID ARCHITECTURES

4.1 Expert Knowledge

Expert knowledge represents the proficiency shown by experts in a particular domain. In the educational domain, expert knowledge is generally available. For example, in [12], expert knowledge was used through a Bayesian network to predict logical reasoning of students. There are many ways to make the expert knowledge usable by a computer. A well-known approach consists in building a rule-based system [5, 2]. In our context, where we only have access to description (in textual form) of each of the levels, we employed two different techniques related to IR (Information Retrieval) and NLP (Natural Language Processing): the Word Movers' Distance (WMD) [8] to compare the meaning of texts, n-grams, and stemming to test whether verbatim contain those specific elements extracted for each level. For each verbatim the WMD calculation provided us with a vector of length 5 where each entry is the similarity between the verbatim and the description of the corresponding socio-moral reasoning skill level. The values range between 0 and 1. For each verbatim, the n-grams provided us with a vector of size 5 where each entry represents the number of n-grams and synonyms of each level found in the verbatim. We finally applied the softmax function to the resulting vector which was added to the one generated by the WMD method.

4.2 Hybrid Architectures

Neural networks (NN) are able to learn any function that maps inputs to outputs. They are particularly salient when there is a lot of data available. Neural networks often perform well when compared to other machine learning algorithms. According to LeCun and his team [9], deep learning allows computational models composed of multiple layers of processing (e.g. neural networks) to learn data representations at multiple levels of abstraction. There exist several deep-learning architectures. However, we will focus here on LSTM and CNN, as they were proven well suited for text classification [17].

Neural Networks are layered graphs with the output of one node feeding into one or many other nodes in the next layer. Their inner architecture makes it difficult to incorporate domain knowledge to the learning process [10]. Our solution is to constrain the model to pay attention to what the expert knowledge says about the current input x . It aims to incorporate the expert's final prediction about the input rather than how the expert knowledge is processed. Since attention [15] is a memory-access mechanism, it fits well in this context where we want the model to have access to the expert knowledge during learning, as its memory. In other words, the DNN will "consult" the expert before taking the final decision. Attention mechanism equips a neural network with the ability to focus on a subset of information. The importance the DNN model will accord to what the expert said is computed (learned) through attentional weights (W_c and W_a). W_c represents the weights measuring the importance of the expert knowledge and the learned features to the final prediction vector. W_a corresponds to the weights calculating the importance of each feature learned by the DNN on each feature of the expert knowledge. Thus, the model will focus on what the expert knowledge says before taking any

decision. By integrating the expert knowledge in the DNN using the attention, the model iteratively processes the a priori knowledge by selecting relevant content at every step. To present our approach, we will consider the LSTM model. However, this can also be applied to the CNN as well. In the original attentional mechanism, the attentional vector is computed from the target hidden state h_t and the input hidden state (which is replaced by the vector representing the expert knowledge). Given the hidden state h_t of the LSTM, and the expert-side context vector c_t^e , we employ a concatenation layer to combine the information from both vectors to produce the attentional hidden state a_t as follows:

$$a_t = \tanh(W_c[c_t^e; h_t]) \quad (1)$$

Please note that h_t could be the hidden state of the last cell of the LSTM or the hidden states of all the cells. For the CNN, h_t represents the output that has been flattened. The attentional vector a_t is then fed through the dense layer of the model to produce the predicted socio-moral reasoning skill level. Now, the expert-side context vector is computed as follows :

$$\begin{aligned} \text{score}(e_s, h_t) &= h'_t \cdot e_s \cdot W_a + b \\ \alpha_{t,s} &= \frac{e^{\text{score}(e_s, h_t)}}{\sum_{j=1}^s e^{\text{score}(e_j, h_t)}} \\ c^e &= \sum_s \alpha_{t,s} \cdot e_s \end{aligned} \quad (2)$$

Where e_s ($s=5$) is the current socio-moral reasoning skill level (normalized) computed from the expert knowledge and $h'_t = \text{softmax}(h_t)$. e_s is a vector of length equals to the number of levels (5 in our case) and each entry represents the predicted probability by the expert that the verbatim belongs to each of the classes. We applied the *softmax* function to h_t to have a fair comparison with e_s . *Score* is a content-based vector [11] which computes the correlation (alignment score) between the expert knowledge and the features learned by the network : how well the expert knowledge and the features learned from data are aligned. The model assigns a score $\alpha_{t,s}$ (vector of size 5 in this case) to the pair of feature at position t and expert knowledge (e_s, h_t), based on how well they match. The set of $\alpha_{t,s}$ are weights defining how much of each expert knowledge features should be considered for each output (final prediction). Our code for this attentional-based a priori expert knowledge is publicly available on github ¹.

5. EXPERIMENTS

In order to empirically evaluate the proposed solution, we investigated six different models: a CNN (cnn-only), an LSTM (lstm-only), a CNN where expert knowledge was concatenated to the features learned (cnn-expert), an LSTM where expert knowledge was concatenated to the features learned (lstm-expert) and finally the proposed models cnn-expert-att (a CNN where expert knowledge was concatenated to the features learned + knowledge-based attention) and lstm-expert-att (a LSTM where expert knowledge was concatenated to the features learned + knowledge-based attention). We used the same parameter initialization for all the models. Adam [6] was used as the optimizer with a learning rate

¹<https://github.com/angetato/Combining-DNN-With-Expert-Knowledge-ToPredict-Socio-Moral-Reasoning-Skills>

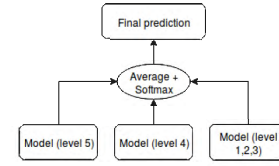


Figure 1: Final model for predicting socio-moral reasoning skill level. Each sub model is specialized for the prediction of levels specified between parenthesis. Each model is either LSTM or CNN combined with expert knowledge and the attentional vector.

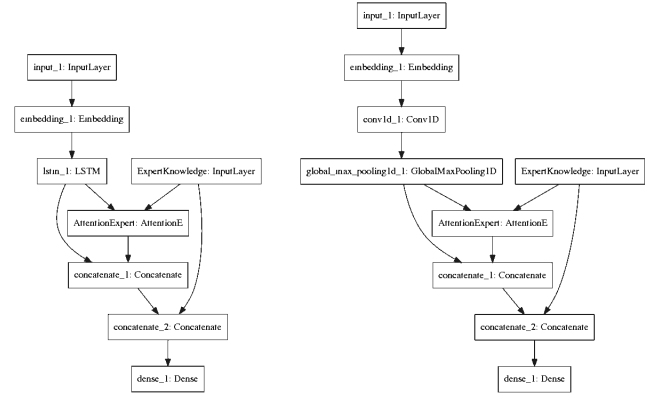


Figure 2: The proposed hybrid architecture using LSTM (left) or CNN (right) for the prediction of socio-moral reasoning level.

set to 0.001. The algorithms were implemented using Keras² and the experiments were done using the built-in Keras models, making only small edits to the default settings. The dataset consists of a benchmark of 731 verbatims (written in French) manually annotated by experts. Verbatims are not equally distributed between classes (unbalancing data).

5.0.1 Proposed Models:

The models take as input the verbatims that have been pre-processed (tokenization, text to sequence, etc.) and vectored. The vectors are then passed to the embedding layer. In figure 2 (left), the embedding vectors are passed to the LSTM layer (note that we have only considered the output of the last cell). The expert knowledge and the output of the LSTM are then passed to the expert knowledge-based attention. The attentional vector is merged with the expert knowledge and the output of the LSTM. The concatenation is passed to the last layer for the prediction. This process is the same for the CNN (see figure 2(right)), except that the knowledge-based attention layer takes as input the expert knowledge and the result of the pooling operation applied to the output of the CNN. To evaluate the added value of this knowledge-based attention, we have considered two similar models (cnn-expert, lstm-expert) to those shown in figures. However, those models do not have the knowledge-based attention layer. They are used as a comparison with the proposed solutions.

5.1 Results

²<https://keras.io/>

All the models have been trained on 80% (with 20% as validation data) of the data and tested on the remaining 20 %. The results are shown in table 1. As we can see, models that take into account expert knowledge perform well for the prediction of classes with few samples compared to others with no expert knowledge (f1score for classes 2, 4, and 5). This is due to the fact that experts do not need data to make decisions because their decisions are based on their own knowledge. This suggests that, incorporating expert knowledge to neural models improves the classification even when the dataset is unbalanced. Also, even if there is not a huge amount of training data, the models are still capable of better generalize on unseen data.

Table 1: Overall precision, recall, f1score and accuracy of all the models.

Models	Precision	Recall	f1-score	Acc
Expert-Only	0.47	0.40	0.38	0.40
cnn-only	0.58	0.53	0.49	0.53
lstm-only	0.42	0.43	0.42	0.43
cnn-expert	0.62	0.62	0.62	0.62
lstm-expert	0.54	0.53	0.51	0.53
cnn-expert-att	0.67	0.65	0.63	0.65
lstm-expert-att	0.59	0.60	0.58	0.60
Final Model	0.72	0.75	0.73	0.75

6. CONCLUSION

We have proposed a model able to predict with over 75% accuracy the socio-moral reasoning skill level based on textual verbatim and a priori expert knowledge. In particular, we proposed a simple but efficient way to integrate expert knowledge into deep neural networks architecture using the attentional mechanism. Our proposed model is also intended to help experts on the annotation of verbatims. Results were very promising. Contrary to the state-of-the-art techniques in text classification, the solution we proposed achieves best results in our context. This is mainly due to the deep structures that can learn useful features from data and also the a priori knowledge that can leverage unbalanced data. Based on theoretical arguments and numerical evidence illustrated in the present work, it is possible to conclude that the a priori expert knowledge provides unique insights to complement and improve DNN data-driven approaches. It is our hope that this paper will spur the interest of our research community, towards developing efficient ways of combining neural networks architecture with theoretical prior knowledge. The work presented in this paper has introduced a simple approach towards this direction, which, however, only represents the tip of the iceberg.

7. ACKNOWLEDGMENTS

We would like to thank the NSERC (National Science and Engineering Council) of Canada and the UQAM's AI Research Center (CRIA) for their financial support.

8. REFERENCES

- [1] M. Beauchamp, J. J. Dooley, and V. Anderson. A preliminary investigation of moral reasoning and empathy after traumatic brain injury in adolescents. *Brain injury*, 27(7-8):896–902, 2013.
- [2] O. Cordón, F. Herrera, and P. Villar. Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base. *IEEE Transactions on fuzzy systems*, 9(4):667–674, 2001.
- [3] G. Coro, P. Pagano, and A. Ellenbroek. Combining simulated expert knowledge with neural networks to produce ecological niche models for latimeria chalumnae. *Ecological modelling*, 268:55–63, 2013.
- [4] J. J. Dooley, M. Beauchamp, and V. A. Anderson. The measurement of sociomoral reasoning in adolescents with traumatic brain injury: A pilot investigation. *Brain Impairment*, 11(2):152–161, 2010.
- [5] J. L. Fisher, S. C. Hinds, and D. P. D’Amato. A rule-based system for document image segmentation. In *[1990] Proceedings. 10th International Conference on Pattern Recognition*, volume 1, pages 567–572. IEEE, 1990.
- [6] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [7] L. Kohlberg. Essays on moral development: The psychology of moral development (vol. 2). *New\brk: Harper & Row*, 1984.
- [8] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger. From word embeddings to document distances. In *International Conference on Machine Learning*, pages 957–966, 2015.
- [9] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [10] H. Lu, R. Setiono, and H. Liu. Effective data mining using neural networks. *IEEE transactions on knowledge and data engineering*, 8(6):957–961, 1996.
- [11] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.
- [12] A. Tato, R. Nkambou, J. Brisson, and S. Robert. Predicting learner’s deductive reasoning skills using a bayesian network. In *International Conference on Artificial Intelligence in Education*, pages 381–392. Springer, 2017.
- [13] A. A. N. Tato, R. Nkambou, A. Dufresne, and M. H. Beauchamp. Convolutional neural network for automatic detection of sociomoral reasoning level. In *EDM*, 2017.
- [14] G. G. Towell and J. W. Shavlik. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1-2):119–165, 1994.
- [15] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.
- [16] A. Zappone, M. Di Renzo, M. Debbah, T. T. Lam, and X. Qian. Model-aided wireless artificial intelligence: Embedding expert knowledge in deep neural networks towards wireless systems optimization. *arXiv preprint arXiv:1808.01672*, 2018.
- [17] C. Zhou, C. Sun, Z. Liu, and F. Lau. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*, 2015.

Identifying bias and underlying knowledge structures in Brazilian National Exam of Students Performance

Mariana S. Oliveira
Intelie & UNIRIO, Rio de Janeiro, Brazil
mariana.oliveira@intelle.com.br

Carlos E. Mello
PPGI - UNIRIO
Rio de Janeiro, Brazil
mello@uniriotec.br

ABSTRACT

The National Exam of Students Performance (ENADE), organised by the National Institute of Educational Studies and Research Anísio Teixeira (INEP), aims to evaluate the performance of undergraduate students in relation to the syllabus, as also skills and competencies acquired during their academic training. The result of this evaluation is considered to be a part of a grade for undergraduate programmes and is also used to assess the general quality of Brazilian higher education. The main goal of the present research is to understand the relationship between the curricular components associated with the questions of the ENADE exam. We investigate the existing bias in the students' answers that may reveal a tendency towards specific fields, possibly resulting in better scores for certain programmes. Open data for the last two years of ENADE for the Information Systems (IS) undergraduate programmes were collected and pre-processed by using data techniques for cleaning, transforming and normalisation. In order to measure the bias in the probability distributions of question answers, the information theory framework was applied. Our experiments revealed structures among questions by conditional entropy. Such structures may suggest underlying relationships between questions, not only for those of the same subject but also for questions on dependent subjects. Our findings show that some questions may give an advantage over at most 15.2% on other questions.

Keywords

entropy, analytics, bias selection, higher education

1. INTRODUCTION

Instituted by Brazilian Federal Law, the National Exam of Students Performance (ENADE) aims to evaluate and obtain a diagnosis on the quality of the undergraduate programmes in Brazil [1]. This evaluation consists of the application of a national exam for a sample of students from all Brazilian undergraduate courses, which verifies their knowl-

edge with regard to the programmatic contents foreseen in the national curriculum guidelines, and the development of fundamental skills and competencies associated with professional qualification.

Since 2004, ENADE has been organised by the National Institute of Educational Studies and Researches Anísio Teixeira (INEP), an Institution linked to the Brazilian Ministry of Education (MEC). The exam occurs once a year, but each area of knowledge is evaluated once every three years.

ENADE provides the following instruments to carry out performance evaluation: (1) a test exam for the first and final year students of each undergraduate programme; (2) an impression questionnaire, to collect students' opinions regarding the physical aspects of the test exam, such as size; and, (3) a socioeconomic questionnaire that focuses on identifying the students' social and economic profile.

The national test exam has two components: (a) questions about general training, common to all areas; and, (b) specific training questions, relative to each knowledge area. The questions regarding general training, as a whole, correspond to 25% of the final exam mark, while the specific questions have a weight of 75%. The score of the specific exam component is split into 15% of the discursive part, which is a simple arithmetic mean of all essay questions; and 85% of the multiple choice part, which is proportional to the number of correct answers, since all the multiple choice questions have the same weight.

The result of ENADE test exams is essential for advancing the Brazilian higher education system as it provides quality indicators for monitoring undergraduate programmes, being used for establishing and evaluating public policies for higher education [7]. Additionally, all ENADE data are publicly available on the INEP open data website, providing a very rich source of information about higher education in Brazil.

In this context, the present work seeks to investigate ENADE data in order to analyse information bias and the underlying knowledge structures from the students' scores. We propose a methodology to address these issues by using an information theory framework. We also investigate, with the use of information measures, possible underlying patterns among curricular components that may introduce bias into the final scores. The intuition behind this idea is to prevent favouring specific curricular components by avoiding mutual informa-

Mariana Oliveira and Carlos E. Mello "Identifying bias and underlying knowledge structures in Brazilian National Exam of Students Performance" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 627 - 630

tion shared between groups of questions. Our hypothesis lies in that the underlying knowledge structures are connected through the questions in such way that the final scores are biased towards particular curricular components, instead of those foreseen in the national curriculum guidelines.

We focused our analyses on the ENADE data of the undergraduate programmes in Information Systems (IS). The choice of IS programmes was due to our expertise and experience, which allow us to analyse the quantitative results from our prior knowledge of the area. Our analyses were carried out on the ENADE data for the 2014 and 2017 editions. These are the two most recent editions with available data¹. In these two years, Information Systems was considered as an independent area by MEC, while in the previous editions it was a part of the Computer Science area.

The research questions addressed by this work are the following: (Q.1) Is there any bias in students' answers that could lead to favouring certain undergraduate programmes over others? (Q.2) Is there any overrated curricular component due to underlying dependence structures across the exam questions? (Q.3) If the bias exists, which would be the areas with this possible advantage?

2. RELATED WORK

ENADE results along different years have been reported as an important quality indicator of undergraduate programmes in Brazil [5]. Studies have reported a certain imbalance in the curricular components behind the questions, favouring certain areas over others in the ENADE tests for the Physics [3] (reconstructs the Reference Matrix of the exam), Psychology [4] (applies a blueprint tool to evaluate the exam content validity), and Physical Education [2] (conducts qualitative approach to study the contents of the exam) undergraduate programmes.

To the best of our knowledge, there are no studies in the literature that aim to identify possible bias in the ENADE test, nor studies that use information theory to measure the amount of information between ENADE's questions.

3. DATA

We collected the microdata, composed of individual records, from ENADE exam conducted in the years 2014 and 2017. Data collection was performed through the INEP website². The microdata file is accompanied by a dictionary of variables, a user manual, the socioeconomic questionnaire and inputs for the R and SAAS programs.

The data of the 2014 edition consists of 154 variables and 481,720 undergraduate students from 33 knowledge areas. The 2017 edition contains 150 variables and 537,436 undergraduate students from 44 areas. These variables concern the course, the student, the institution of higher education, the questionnaires that come with the exam, and the students' performance.

¹ENADE's was first applied for the Information Systems undergraduate programmes in 2005; since then it has been reapplied every three years.

²<http://inep.gov.br/microdados>

From the both databases, we extracted the information of the undergraduate students of Information Systems Education who answered the multiple choice questions of the specific component. The discursive questions were not considered, there being a lot of subjectivity in the score.

The variable of interest extracted is a sequence of characters, per student, representing the score of each multiple choice question within the specific component. Each character is a code, 0 for get it wrong and 1 for score. Codes 8 or 9, showing respectively, annulment by the commission and annulment by the index of discrimination, were disregarded. We also ensured that all students answered all questions removing observations where the selected variable was zero or different in size from the total number of questions. After that, the 2014 edition was composed of 13,253 students; that of 2017, of 11,980 students.

The structure of the data was better organised to facilitate analysis. A Boolean matrix was constructed from the variable of interest, with rows representing the students and the columns corresponding to each question. The dimension of 2014 edition matrix was $13,253 \times 19$, while for 2017 was $11,980 \times 22$. Although the multiple-choice specific questions of ENADE are 27 in all, in both editions there were voided questions.

4. METHODS AND METHODOLOGY

Our approach for investigating the possible dependencies of knowledge and competence structures in the ENADE exam was based on the Information Theory framework. The motivation was the strong mathematical and statistical theory [6] that supports the empirical results. The principles of Information Theory have been widely applied to many different fields such as telecommunications, physiology, linguistics, and physics [8]. Our interest is particularly in information measures and dependencies.

4.1 Information Theory

Information Theory studies the quantification, storage and communication of information. It was originally introduced in a 1948 paper by Claude E. Shannon titled *The Mathematical Theory of Communication*, in which the author was interested in the amount of information that a communication channel would be able to transmit.

The main measure of information theory is the entropy ("H"). Entropy is a quantification measure of uncertainty [8], given by the following equation³:

$$H = - \sum_i p_i * \log(p_i), \quad (1)$$

where p_i indicates the probability of each result of a discrete random variable. Note that entropy corresponds to a function of the distribution of the random variable; referring only to its probabilities, regardless of the values it takes.

In addition to entropy, information theory also has measures that deal with a pair of random variables. The joint

³In the present work, we used log base 2. However, other bases could be used, such as 3, 4, 5, 6, 7 or 10. The base change can be made by $H_b(X) = (\log_b a) * H_a(X)$.

entropy $H(X, Y)$ quantifies the amount of information associated with the random variables X and Y taken together; the conditional entropy $H(X|Y)$ denotes the amount of information that a random variable X holds regardless of the amount of information held by Y . Finally, the mutual information $I(X; Y)$ describes the amount of information that a random variable X contains about the other variable Y [8].

We adopted the conditional entropy as being the measure of dependence between two variables, given by equation 2⁴:

$$H(X|Y) = \sum_{y \in \gamma} p(y)H(X|Y = y). \quad (2)$$

4.2 Proposed Methodology

We have proposed a methodology consisting of the steps described next and in Algorithm 1.

Algorithm 1: Pseudocode of proposed methodology

input \leftarrow Students' answers from ENADE;

$M \leftarrow$ Conditional Entropy Matrix(*input*);

$M \leftarrow$ Min Max Normalisation(M);

$G \leftarrow$ Graph From Adjacent Matrix(M);

$G.nodes \leftarrow$ Entropy($G.nodes$) ≥ 0.99 ;

$G.links \leftarrow G.links < 0.4$;

output $\leftarrow G$;

The thresholds 0.99 and 0.4 are empirical, by filtering for very high entropy questions and for an acceptable minimum value of conditional entropy between questions.

(a)Conditional entropy matrix: we developed a function for calculating the conditional entropy of each pair of random variables. This allows us to look into the relationship between questions of different areas and identify possible areas that significantly contribute to the score of others. As input, this function receives the Boolean matrix from the pre-processing phase and as output, generates a square matrix, question by question (dimensions 19×19 for 2014; 22×22 for 2017), where each element in row i of column j corresponds to the conditional entropy of question q_j , since question q_i is known. Data was normalized by Min-max normalisation⁵ [9], since the values were very close to each other. The diagonals were taken as 1.

(b)Graph construction: the conditional entropy matrix was taken as an adjacency matrix to construct a directed graph with the objective of simplifying the visualisation of dependencies. Each node of the graph represents a question and each relationship weight represents the conditional entropy between them.

(c)Node filtering: we select the questions with the highest entropy values (between 0.99 and 1). This prevents possible bias in our analysis generated by extremely easy or extremely difficult questions, given that the resulting questions have approximately 50% chance of success; therefore, they can't be considered easy (it's already known that the

⁴This equation can be decomposed into equation $H(X|Y) = - \sum_{y \in \gamma} \sum_{x \in \chi} p(x, y) \log p(x|y)$, which is the one used in the methodology.

⁵Min-max normalisation is given by $x' = \frac{x - \min(x)}{\max(x) - \min(x)}$.

vast majority of students are going to score) or difficult (it's known that the vast majority of students are going to get it wrong). For 2014, questions 10 and 13 were not included since they only relate to one question each, being isolated from the others. In 2017, there were questions with high entropy not selected because we also apply a relationship filter, to be discussed next.

(d)Relationship filtering: a filter was applied to remove the relationships holding the lowest conditional entropy values (between 0.4 and 0). Thus, we are asserting the force of the influence that the random variable Y exerts on X . For instance, the question X solely holds an entropy close to 1. But its entropy is close to 0 after question Y is known, even when the latter question has exactly the same profile as question X . The final graph only contains high entropy nodes and low conditional entropy relationships.

Table 1: Advantage quantification per question from the 2014 edition. The highest gains are highlighted in yellow.

X	Y	$H(X Y)$	$P(X=1)$	$P(Y=1)$	$P(X=1 Y=1)$	%G
24	27	0.056	0.467	0.527	0.501	7.4%
24	28	0.056	0.467	0.557	0.499	7.0%
15	28	0.084	0.469	0.557	0.501	6.9%
24	25	0.087	0.467	0.552	0.498	6.9%
15	32	0.126	0.469	0.515	0.502	7.2%
24	32	0.141	0.467	0.515	0.499	7.0%
15	11	0.162	0.469	0.482	0.503	7.4%
24	11	0.196	0.467	0.482	0.499	6.9%
15	25	0.219	0.469	0.552	0.497	6.0%
32	27	0.266	0.515	0.527	0.562	9.2%
11	27	0.272	0.482	0.527	0.554	14.8%
27	28	0.279	0.527	0.557	0.583	10.6%
25	27	0.323	0.552	0.527	0.596	7.9%
25	11	0.331	0.552	0.482	0.599	8.6%
24	15	0.333	0.467	0.469	0.493	5.7%
28	25	0.351	0.557	0.552	0.617	10.8%
15	24	0.359	0.469	0.467	0.495	5.7%
25	32	0.367	0.552	0.515	0.593	7.5%
28	27	0.399	0.557	0.527	0.616	10.6%

Table 2: Advantage quantification per question from the 2017 edition. The highest gains are highlighted in yellow.

X	Y	$H(X Y)$	$P(X=1)$	$P(Y=1)$	$P(X=1 Y=1)$	%G
35	27	0.049	0.548	0.508	0.631	15.2%
19	27	0.053	0.46	0.508	0.516	12.2%
23	26	0.059	0.455	0.491	0.506	11.0%
23	35	0.109	0.455	0.548	0.498	9.3%
19	35	0.212	0.46	0.548	0.505	9.7%
30	27	0.26	0.547	0.508	0.618	13.1%
19	26	0.29	0.46	0.491	0.506	10.0%
27	35	0.313	0.508	0.548	0.585	15.2%
23	30	0.314	0.455	0.547	0.487	7.0%
26	35	0.333	0.491	0.548	0.548	11.5%
23	19	0.352	0.455	0.46	0.49	7.7%
35	26	0.383	0.548	0.491	0.611	11.5%

5. RESULTS AND DISCUSSION

Our results show that the two 2014 and 2017 ENADE editions presented groups of highly connected questions from strongly related areas of knowledge. In both cases, the conditional entropy allowed us to highlight the underlying dependence structures of the curricular components behind the questions. Such structures can be represented by a graph, in which the questions are the nodes and the edges indicate the dependence relationships.

Figure 1a depicts the resulting graph based on the 2014 edition of ENADE. You will see that questions 11, 15, 24, 25, 27, 28 and 32 are nearly all directly or indirectly related to

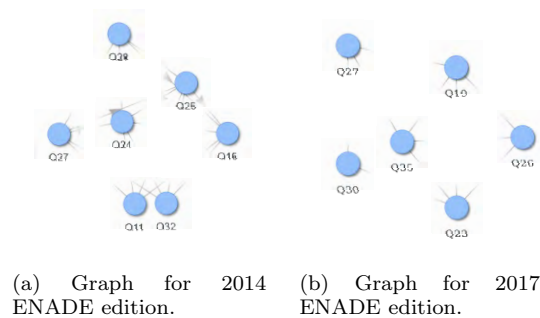


Figure 1: Graphs for 2014 and 2017 ENADE editions.

the area of Software Engineering, the only exceptions being questions 11, 15, and 28. Question 11 is about Probability and Statistics, but is quite intuitive, demanding only logical reasoning, while the question background still relies on software operation. Besides, this question has more influence on the others than the opposite. Question 15 is about Fundamentals of Systems and Databases, requiring awareness of differences between Online Transaction Processing and Online Analytical Processing systems subject (very related to Software Engineering). Finally, Human-computer interaction is the main subject of question 28, addressing Usability with Software Requirements Specification (strongly dependent on Software Engineering). We may therefore conclude that our methodology was able to identify the questions in the 2014 ENADE edition that require knowledge and competences related to Software Engineering.

In order to highlight the importance of the dependence relationships, Table 1 describes, for each edge of the graph in Figure 1a, the percentage of advantage gain (G) for scoring a question, given the correct answer of another. This measure consists of the percentage of the conditional probability $P(X|Y)$ over the prior $P(X)$. As one should note, question 27 influences the largest number of questions (a total of 5) and also provides the greatest advantage gains (a total of 3) for many questions. As a result, students who correctly answer these questions are more likely to get, at least, three other questions right. Undoubtedly, there are underlying relationships that tie question 27 to the others. This leads us to support the hypothesis of bias in the students' responses, favouring the area of Software Engineering and the ones closely related.

The same scenario can be clearly verified in the 2017 edition. Consequently, there is also a group of interconnected questions with very close contents. Note that questions 19, 23, 26, 27, 30 and 35 are all highly associated to Software Development area, even though questions 19 and 27 are not directly connected to it. The content behind question 19 is Graph theory and its area of knowledge is Operations research, a discipline that deals with the application of advanced analytical methods to solve problems and help to make better decisions. Although this is a mathematical approach, this area is still very tied to programming (the programming area is one of the essential bases in Software Development); Question 27 is about Knowledge management, yet the question addresses the use of Tacit and Explicit Knowledge in the different phases of Software Development.

Similarly to Table 1, Table 2 was reproduced for the 2017 edition graph. It reports the influence of questions 26, 27 and 35 on each other, and on others quite clear. Scoring question 35 leads to higher chances of scoring 26, which in turn increases the chances of correctly answering questions 23 and 19; and question 27, brings an advantage for scoring questions 19 and 30. Overall, by answering question 35 correctly, there will be four other questions more likely to score. Thus, we believe that there is also a bias in the 2017 edition, favouring Information Systems programmes, or student profiles, with focus on Software Development.

6. CONCLUSIONS

Our methodology has successfully identified potential knowledge dependence structures through mutual information between questions. The adoption of information measures has proven to be very helpful, providing good level of interpretation to reveal underlying dependence structures. However, a limitation of our work is to deal with deeper and more complex underlying structures, which require multilevel analyses and different strategies to visualize and evaluate relationships between questions.

7. ACKNOWLEDGMENTS

The authors would like to thank FAPERJ - Foundation for Research Support of Rio de Janeiro State (E-26/201.670/2017).

8. REFERENCES

- [1] Ministério da educação. <http://inep.gov.br/enade>. Accessed: 2019-02-23.
- [2] E. Coelho Bié, M. Janaína Lustosa Souto, D. Martins Braga, L. Araujo de Sousa, and J. Airon Pontes Jr. Categorização dos itens das provas de licenciatura em educação física do enade. 10 2016.
- [3] J. P. de Castro Costa and M. I. Martins. O enade para a licenciatura em física: Uma proposta de matriz de referência. *Revista Brasileira de Ensino de Física*, 36(3):3401, 2014.
- [4] G. R. de Jesus, R. M. de Lima Rêgo, and V. V. de Souza. Evidências de validade de conteúdo da prova de psicologia do enade. *Estudos em Avaliação Educacional*, 29(72):858–884, 2018.
- [5] C. M. Griboski. O enade como indutor da qualidade da educação superior. *Estudos em avaliação educacional*, 23(53):178–195, 2012.
- [6] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer New York Inc., 2001.
- [7] M. M. Polidori, C. M. Marinho-Araujo, and G. B. Barreyro. Sinaes: perspectives and challenges in evaluation of brazilian higher education system. *Ensaio: Avaliação e Políticas Públicas em Educação*, 14(53):425–436, 2006.
- [8] J. C. Principe. *Information Theoretic Learning: Renyi's Entropy and Kernel Perspectives*. Springer Publishing Company, Incorporated, 1st edition, 2010.
- [9] C. Saranya and G. Manikandan. A study on normalization techniques for privacy preserving data mining. *International Journal of Engineering and Technology (IJET)*, 5(3):2701–2704, 2013.

Validating The Myth of Average through Evidences

Praseeda
IIIT Bangalore
26/C, Electronics City
Bengaluru 560100, India
praseeda@iiitb.org

Srinath Srinivasa
IIIT Bangalore
26/C, Electronics City
Bengaluru 560100, India
sri@iiitb.ac.in

Prasad Ram
Gooru Inc, 350
Redwood City
CA 94065, USA
pram@gooru.org

ABSTRACT

Most formal educational practices based on the classroom, imparts skills and knowledge at scale, by adopting the model of a *factory*. Here, the focus is on creation of formal processes, mass production of measurable outcomes, and standardization. Curricula and educational practices are designed for a hypothetical “average” student, having “average” abilities. However, recent advances in individualization have found vast discrepancies between individual traits and group averages. Designing models based on group averages, are usually ineffective when the *individual* is the target beneficiary. This research proposes *Evidence Based Competency Model* as a mechanism for providing individualized learning experiences. Here, the term *evidence*, refers to informal data generated by the learner, pertinent to the learning process. Individual models are built for each learner, based on their learning activities. These models are clustered to observe similarity in learning patterns among the individual learners. This study also shows significant variability from the “average model”, validating *The Myth of Average*.

Keywords

Evidence, Average Model, Individualization, Evidence Based Competency Model, Learning Process

1. INTRODUCTION

The objective of education is to create empowered individuals that are capable of problem solving, upholding their individuality, and possessing an ability to acquire relevant competencies in their area of interests. Standardization of learning practices based on the classroom, has lead to uniform teaching and assessment practices without regard to how disparate individuals behave, learn, develop and apply their knowledge.

The classroom model is able to provide pedagogical solutions at scale by addressing the needs of a hypothetical “average” individual. However, research on individualization

have shown that, as the number of dimensions of concern increase, the probability of finding an individual who is average on all dimensions rapidly diminishes to zero. This is popularly called *The Myth of the Average* [1]. A uniform model based on statistical averages does not capture the patterns of variability among individual learners [2]. Each individual has different interests, disposition, contexts and styles to learn different topics.

One of the emerging approaches towards standardizing pedagogy models, is to focus on *learning outcomes*. The Outcome Based Education (OBE) [3] model focuses on learner’s ability to produce specific, measurable outcomes as part of the learning process. The emphasis on visible outcomes, discounts the holistic nature of education comprising of a number of tacit elements and OBE is also considered to be strongly rooted in behaviorist learning practices, that are incompatible with other learning cultures like constructivist education [4].

In this work, we show that data generated during the learning process, referred to as *evidence*, can be used to reason about the underlying competency. We also show that one single average model to predict the state of the competency cannot be used and we need to build separate models for learners, hence validating the Myth of Average, discussed in more detail, in [5].

The use of activity data, rather than assessments and outcomes, for determining learner’s latent competency levels, have been addressed for specific activities in [6, 7]. The focus here has been to correlate different types of learning activities like watching video or learning by doing activities, to their implication on learning [8].

In this work, we don’t focus on any specific learning activity and follow a generic approach to collect any kind of learning data and use machine learning techniques to correlate characteristics of activity data with data from outcomes and formal assessments. We focus on activities involving learners consuming resources like videos, text books, articles, documents in the current implementation of the model.

2. EVIDENCE MODELING

The term “evidence” is contrasted with “outcomes” as follows: outcomes refer to assessment data collected from formal testing environments, where the learner is fully cognizant of being assessed and has explicitly prepared for the

Praseeda, Srinath Srinivasa and Prasad Ram "Validating the Myth of Average through Evidences" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 631 - 634

same. In contrast, evidence pertains to data collected on an implicit, continuous basis on any activity of the learner pertaining to the competency in question. Existing methods of determining the underlying competency of a learner through formal assessments and visible outcomes, has its own issues [3, 4]. There is a need for models that uses evidences, based on the learner’s activities and maps these learners to their competencies.

We propose a model that explores the activity data of the learners, generated while achieving the competency. The data, collected implicitly in a continuous fashion, is modeled to map the learners to their competencies.

Evidences can be considered analogous to an observation that an individual makes, in the classroom through interaction or silent observation of reactions, which contains significant insights. An offline system like a classroom is unable to gather data for each individual during the learning process, but when learning happens in a Technology Assisted Learning Environment (TALE) [9], data is continuously captured by the platform. Technology augmentation can happen in various ways – like sensors and RFID tags to record attendance, recording and analysis of students’ classroom activities like questions, discussions, etc. We want to focus on the evidences, so that we don’t have to rely only on the formal assessments to determine the competency of the learner.

A competency model based on evidences can also be used to identify anomalous cases where the outcomes state that the individual has the competency while the evidence indicates a lack of competency, or vice-versa. A teacher can only analyze such cases and address them appropriately. The model based on evidences, acts as a way of aiding the teacher in teaching and evaluating process.

Newer learning domains like training drivers to learn a particular language that would improve their communication skills, may not have standardized competency models to aid in pedagogy and assessments. In such cases, models based on evidences would help to map the learners to their competencies.

The main challenge is to create an evidence model for collecting data, and to argue for the completeness of the evidence. Learning may happen outside of the evidence gathering, and different kinds of learning activities may require different kinds of evidences to reason about them.

In order to address the above challenge, we adopt a least-biased model for evidence modeling, and treat each form of evidence as equally important in the input feature vector. All forms of evidences collected are then given as input to a machine learning algorithm to find the best possible indicators for the outcomes based on assessment scores.

3. EXPERIMENTS AND INITIAL RESULTS

The activity data used to build models is collected from a large, open online learning platform, implemented across several schools in the US, *Gooru.org*¹. The platform has aggregated open learning resources for various courses, pro-

vided by content creators, curators and instructors. The learning resource can be a document, video, audio, puzzles or any content used to obtain the competency. Learners enroll to various courses.

A course is organized into several competencies, where a competency is seen as the basic unit of learning. Each competency may have several learning resources mapped to it. Students consume learning resources and whenever they are ready, give assessments to earn a score. Instructors evaluate the assessments and provide their feedback in the form of scores. The scores indicate the status of the learner with respect to the competency. The learning resources are mapped to competencies for various courses like Maths, Science, English and Social Science in the K-12 curriculum².

The activity data collected for a (learner, competency) pair, is divided into collections and assessments. Collections are resources used during the learning process, while assessments act as indicators of learning. In the platform, a learner is said to have achieved a “completed” status for a competency if one gets more than 80% in the respective assessments. The platform does not award a “fail” status to the learner. The learner keeps attempting assessments multiple times until the learner gets 80% or more. The status is then set to “completed”, else the status is marked as “in-progress”.

Whenever a learner consumes a resource to learn a concept, an event is logged in the system with the details of the resource, time of the event, learner details and type of event (started consuming the resource). The same process is repeated when the learner finishes consuming the resource with a stop event. The events are captured for all the courses. Individuals consume various resources mapped to the same competency and at the end give assessments to get a particular status for that competency.

Using these activity data we have built a model to determine the outcome of the competency based on the evidences.

3.1 Experiment 1

We built a Support Vector Machine (SVM) [10] classifier to test a hypothesis. Our hypothesis is as follows: H_t : *The time spent on learning resources, the total number of resources consumed by learners, can predict the outcome for the underlying competency.*

The features identified are total time spent on resources, average time spent on resources and number of resources used for acquiring the corresponding competency. The users were given a completed or in-progress status based on their assessment scores. That serves as the ground truth for our model.

The dataset has 28000 (user, competency) pairs with their corresponding evidences. This data was divided into 80-20 split randomly for training and testing the SVM model respectively. We built a single SVM model for all learners and their competencies in all courses. We classified the data using linear, polynomial, sigmoid and radial basis kernel function. The data was not linearly separable. The radial basis

¹<http://gooru.org/>

²<https://en.wikipedia.org/wiki/K-12/>

kernel function was found to be the best kernel function to classify the data in terms of Accuracy, Precision and F1 score as shown in Table 1.

Kernel functions	Accuracy	Precision	F1 score
Radial basis	82.43%	68.79%	51.13%
Linear	78.39%	53.65%	40.30%
Polynomial	78.83%	56.25%	37.91%
Sigmoid	68.84%	30.68%	30.37%

Table 1: Performance metrics comparing the kernels

The accuracy measure of the SVM model using the radial basis function states that using total time spent, average time spent on the resources and number of resources, we can significantly predict the outcome of the underlying competency. The outcomes are determined by the scores of the assessments, which was not used as feature to build the models. This shows that evidence can be used as an alternate way to model the outcome of the underlying competency.

The same model was made to classify all the competencies of a random individual learner, the accuracy of the average SVM model varied between 30% to 100% for different learners. The “average learner” model computed above, was not effective in determining the outcome of a competency of an individual learner. This indicates that we cannot use one single aggregated model on all individuals. Models must consider personalization i.e., we need to build individual models for each users and aggregate the models based on common properties.

To do this, we require significant amount of data for each learner. So, instead of aggregating time spent on resources at a competency level as total time, we looked at time spent for each resource and aggregated the data separately for each learner in the next experiment.

3.2 Experiment 2

In this experiment, we used data from each activity event and modeled the time spent on each resource mapped to a particular competency. Using the start and the stop time, we computed the time spent on each resource and we also observed that some individuals consumed the same resource again. Based on this, we formed our second hypothesis: H_u : *There is a large variance among the individual models built for each learner based on their learning activity data.*

We wanted to observe the consumption of resources and the time spent on those resources, could predict the outcome of the corresponding competency for that individual learner. We also wanted to observe if the models built for each individual learner had common properties and could be merged to a single model or there is a large variance among them.

The amount of time spent on each resource is stored as a vector for each (learner,competency) pair, the length of the vector is the number of resources and the order of the vector tells the order in which the resources were consumed. The length of the vector varied for each (learner,competency) pair. To build individual learner models and compare them, we require equal number of features for all pairs of (learner, competency).

To achieve this we transformed the time spent on resource vector, to a matrix in the following way. We computed the range of total time spent on (resource,competency) pair by all learners. This value varied from 10 seconds to 46000 seconds. After observing this distribution, we decided to have a time frequency of 100 seconds and computed the cumulative sum of resources consumed by the learner at each frequency i.e. 100th second, 200th second etc. We populated 460 time frequency columns. We arrived at this time frequency value of 100 seconds by dividing the maximum total time and time frequency.

For example, if the learner has consumed 2 resources for a competency code “3”, spending 90 seconds on first resource and 170 seconds on second resource then the column 0(time_100) will have the value 1, column 1(time_200) will have the value 1 as the learner has not finished consuming the second resource by 200 seconds, column 2(time_300) has the value 2 and rest of the columns till column 459(time_460) will have the value 2 indicating that the user consumed maximum 2 resources. Fig. 1, shows the subset of the data passed to the model for a single user, where code refers to competency code and rest of the columns are the evidences for that competency code for a single learner. The row containing code value 3, shows the corresponding result of example mentioned above. The features also includes normalized total time spent on resources and normalized average_time spent on those resources for a particular (competency, learner) pair.

Code	Total_time	Average_time	0	1	2	3	4	5	6	7	...	451	452	453	454	455	456	457	458	459	status
0	0.588666	0.998084	0	0	0	0	0	0	0	0	...	2	2	2	2	2	2	2	2	2	0
1	0.348463	0.128140	0	0	1	1	1	1	1	4	6	...	9	9	9	9	9	9	9	9	0
2	0.000000	0.000000	1	1	1	1	1	1	1	1	...	1	1	1	1	1	1	1	1	1	1
3	0.060333	0.100575	1	1	2	2	2	2	2	2	...	2	2	2	2	2	2	2	2	2	1
4	0.294333	1.000000	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1	1	1

Figure 1: Data passed to the classifier with cumulative frequency of resources with respect to time, total_time and average_time as features.

SVM model with linear kernel was used to classify the data. Learners who had more than 30 competencies in any course and with any status (completed or in-progress), were selected for analysis. There were 42 learners who satisfied that criteria and individual SVM models were built for those learners. Individual learner’s data was divided into 80-20 split randomly for training and testing respectively. The models for each learner gave an accuracy between 80% to 100%. The accuracy was calculated from the confusion matrix generated for each individual model according to the formula mentioned in ³. Each model generated a coefficient vector of length 462. We also built a single model comprising the activity data all these 42 learners and called this the “average-model”, as detailed in Experiment 1, for comparison.

All the 43 vectors of coefficients for the 43 models populated were clustered using the K-nearest neighbor clustering algorithm⁴ to observe any similarity among these models.

³https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers

⁴https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm/

The Euclidean distance between the data points (models) was used as a measure to cluster the models. To find the appropriate number of clusters, Elbow method⁵ was used to find the optimal number of clusters. The summation of distance between that specific cluster against the cluster centroid was computed and plotted for various number of clusters, between 1 and 43 where 43 was the total number of models. The graph in Fig. 2 shows the variance in the sum of the squared distance between clusters against the number of clusters.

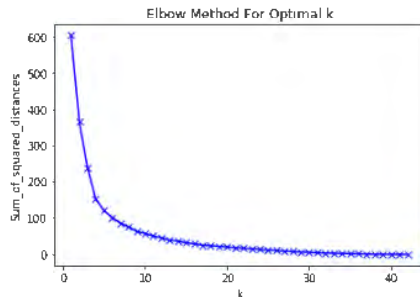


Figure 2: Elbow method showing the variance of number of clusters and their distances

Using the variance in the Elbow method, we found the optimal number of clusters to be 6. This says that the 43 models including the average model can be clustered into 6 clusters, and there are some common properties among these models.

Fig. 3 shows the distribution of 43 different models including the average model into 6 different clusters. The blue point refers to individual learner models and the red point indicates the “average learner” model.

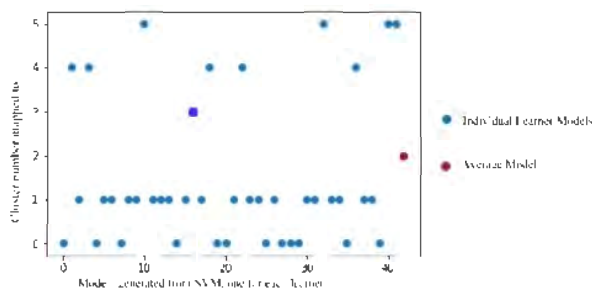


Figure 3: Distribution of individual learners models and average model into 6 clusters

We see that the individual models don’t cluster with the average model. For this dataset, the average model does not represent any individual learner. This validates our hypothesis that there is large variance among the individual models built for each learner based on their learning activity data and hence validates “The Myth of the Average”. It also shows that we need not build one model for each individual which would make it extremely difficult to map new

learners to their competencies. We find that there are some common properties among the learners which can be used to find the best model for each learner. We need to determine those common properties in the future and provide better learning experiences to each individual learner.

4. CONCLUSION

We proposed an Evidence Based Competency Model, that uses data generated during the learning process by learners to find the outcome of the competency. The experiments shows that having a single average model can be tuned to get a higher accuracy, but does not represent any individual. This presses the need to consider individual learner’s variance and not rely on statistical averages to map learners to their competencies. This also says that we cannot provide each learner the same learning experience and cannot validate their underlying competencies through uniform evaluation mechanisms like outcomes. This research shows initial results towards that direction.

5. ACKNOWLEDGEMENTS

The authors would like to acknowledge and thank the contributions of the project associates Thrivikram Mudunuri, Juhi Singh, Naman Dosi and Kartik Gupta.

6. REFERENCES

- [1] Todd Rose. *The end of average: How to succeed in a world that values sameness*. Penguin UK, 2016.
- [2] L. Todd Rose, Parisa Rouhani, and Kurt W. Fischer. The science of the individual. *Mind, Brain, and Education*, 7(3), 2013.
- [3] Maureen Tam. Outcomes-based approach to quality assessment and curriculum improvement in higher education. *Quality Assurance in Education*, 22(2):158–168, 2014.
- [4] Laurie Brady. Outcome based education: a critique. *The Curriculum Journal*, 7(1):5–16, 1996.
- [5] Praseeda, Srinath Srinivasa, and Prasad Ram. Validating the myth of average through evidence based competency model. *Gooru Tech Report*, 3(2), 2019.
- [6] Stephen E. Fancsali, Guoguo Zheng, Yanyan Tan, Steven Ritter, Susan R. Berman, and April Galyardt. Using embedded formative assessment to predict state summative test scores. *LAK ’18*. ACM, 2018.
- [7] Mingyu Feng, Neil T. Heffernan, and Kenneth R. Koedinger. Predicting state test scores better with intelligent tutoring systems: Developing metrics to measure assistance required. Springer Berlin Heidelberg, 2006.
- [8] Kenneth R. Koedinger, Jihee Kim, Julianna Zhuxin Jia, Elizabeth A. McLaughlin, and Norman L. Bier. Learning is not a spectator sport: Doing is better than watching for learning from a mooc. *L@S ’15*, pages 111–120. ACM, 2015.
- [9] J.D. Fletcher. Evidence for learning from technology-assisted instruction. *Technology Applications in Education: A Learning View*, pages 79–99, 2003.
- [10] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

⁵[https://en.wikipedia.org/wiki/Elbow_method_\(clustering\)](https://en.wikipedia.org/wiki/Elbow_method_(clustering))

ATC Framework: A fully Automatic Cognitive Tracing Model for Student and Educational Contents

Yanjun Pu

State Key Laboratory of Software Development

Environment

Beihang University

Beijing, China

puyanjan@nlsde.buaa.edu.cn

Wenjun Wu, Tianrui Jiang

State Key Laboratory of Software Development

Environment

Beihang University

Beijing, China

{ wwj,jiangtianrui }@nlsde.buaa.edu.cn

ABSTRACT

Intelligent Tutoring Systems for online learning need to design their cognitive and student models to analyze every student's dynamic knowledge state and factorize the cognitive process for solving each online exercise into the latent skill sets. In this paper, we present the Automatic Temporal Cognitive (ATC) model, a unified and integrated framework to automatically discover a multiple-dimensional cognitive model and formulate a student model over longitudinal student data. This framework enables us to trace the dynamic change of multi-dimensional skills including skill improvement and forgetting for the student learning process. Moreover, based on the framework, we can automatically build the cognitive model through student performance data to describe the latent skill vector (aka Q-matrix) for educational content. Experimental results confirm that our unsupervised approach is better than the traditional method in public datasets.

Keywords

Cognitive model; Student model; ATC framework, Skills dimension determination.

1. INTRODUCTION

Over the past decade, intelligent tutoring becomes increasingly important in online education to offer personalized and adaptive learning experience for massive scale of students. An essential question about building intelligent tutoring systems for online education is how to develop the cognitive model of educational contents and student model from student response data in online education platforms[1]. A typical application scenario is personalized learning path recommendation where an intelligent tutoring system must infer the latent multi-dimensional skill vectors of every student from a temporal sequence of student interactive events. It often needs both cognitive model and student model to trace dynamic change in a student's skill and plan the personalized learning route.

Traditionally, the cognitive model and student model of an intelligent tutoring system must be designed in multiple steps: First, domain experts specify an original cognitive model in the form of Q-Matrix[2, 3] through the process of Cognitive Task Analysis (CTA)[4]. Second, based on the Q-Matrix that specifies the association among latent skills and education contents, researchers

can design the student model using different kinds of modelling frameworks such as IRT[5], Bayesian Knowledge Tracing (BKT)[6]. Third, the parameters of the student model must be fitted with student response data. The above cycle has its limitations because the subjective specification of the Q-matrix may not be reliable and could possibly result in insufficient model fitting and inaccurate estimation of the latent cognitive skills of students.

Recently, data-driven methods have been proposed to automatically or semi-automatically discover the cognitive models. They can be regarded as the further step to validate and refine the expert-designed cognitive model. But current methods for discovering cognitive models are restricted in that they cannot handle longitudinal data. On the other hand, the popular student models such as Item Response Theory (IRT) model and Learning Factors Analysis (LFA)[7] only evaluate student ability at the moment when they take examinations or exercises. They don't consider the dynamic change in student ability with their interactions with education contents. Knowledge Tracing (KT) can model students' changing knowledge state during their skill acquisition process. But the conventional KT model doesn't support multi-dimensional skill model and must work with other algorithms to perform Q-Matrix discovery.

In summary, the existing frameworks for cognitive models and student models are not well integrated to effectively deal with multi-dimensional and longitude learning data generated by personalized online learning systems. In this paper, we propose the Automatic Temporal Cognitive Tracing framework aiming to build the following models from student learning data: (1) dynamic multi-dimensional student model with the forgetting factor (2) multi-dimensional cognitive model. The student model can describe the forgetting factor in the multi-dimensional skill levels during the student learning process. The cognitive model can automatically discover the skill set (aka Q-matrix) that include the latent skill vector for educational content. The ATC framework seamlessly integrates both models and enables us to estimate dynamic student ability and Q-matrix. The major contributions of our paper include:

- (1) **Propose a dynamic student model including students' skill acquisition and skill forgetting.**
- (2) **A fully automatic approach to discovering a cognitive model for educational contents.**

The paper is organized as follows: Section II describes the current research on student model and cognitive model. Section III introduces the automatic temporal cognitive framework in detail. In section IV, we introduce the datasets for experiments. Section V shows experimental results in different datasets and situations. Finally, sections VI gives the conclusion and discusses future work.

Yanjan Pu, Wenjun Wu and Tianrui Jiang "ATC Framework: A fully Automatic Cognitive Tracing Model for Student and Educational Contents" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 635 - 638

2. RELATED WORK

2.1 Conjunctive Factor Model and Automatic Cognitive Analysis

Researchers have introduced performance factor analysis models such as Conjunctive Factor Model(CFM) [7] to build student model and support automatic cognitive discovery[8]. CFM is essentially a parametric IRT model to model the item responses given the latent Q-Matrix. Eq(1) defines the probability for student i to get item j right, where θ_i represents coefficient for proficiency of student i , β is skill difficulty coefficient vector, γ is skill learning rate coefficient vector and T_{ik} is the number of practice opportunities for student i to learn the skill k .

$$p_{ij} = \Pr(Y_{ij} = 1 | \theta_i, \beta, \gamma) = \prod_{k=1}^K \left(\frac{e^{\theta_i + \beta_k + \gamma_k T_{ik}}}{1 + e^{\theta_i + \beta_k + \gamma_k T_{ik}}} \right)^{q_{jk}} \quad (1)$$

The Q-Matrix in the CFM model has to be explicitly defined before it can be fitted in learning data. To extend the capability of the CFM, Matsuda[8] introduces a machine learning workflow for automatic discovery of skill models from online course data. The framework consists of three major steps: feature extraction, cognitive skill model construction and model search. The step of feature extraction attempts to calculate a candidate Q-matrix using a combination of NMF and K-means clustering method. The step of model construction and model search work together to make further exploration on the skill dimension and find the best Q-matrix and student model at the same time. The step of model construction proposes new Q-matrix candidate by splitting and merging skills in the initial Q-matrix, and the step of model search tries to fit the data with the Q-matrix candidate and student model in the form of CFM or other model families.

The enhanced CFM model with cognitive discovery still has its limitation. As an extension of classic IRT model, the CFM model doesn't describe the dynamic change of the student proficiency. Although its parameters contain the skill learn rate, it doesn't explicitly model the improvement of a student's skill vectors or the forgetting factor in his learning process.

2.2 Knowledge Tracing and Multiple-Skill Extension

Bayesian Knowledge tracing (BKT) was presented by Corbett and Anderson [6]; it uses a Hidden Markov Model (HMM) to track students' knowledge states over time. It divides the target knowledge into several skills, and each skill designs several assessments. But an issue with the conventional BKT model is that it can only track student mastery of single cognitive skill and doesn't support assessment of item difficulty.

Recent research development on BKT models focus on multiple-Subskill Extension in BKT modelling. Brenes[9] extended the KT

model and proposed Dynamic Cognitive Tracing which is a fully automatic approach to discover a cognitive model and student model of longitudinal student data. In his later work, Gonzalez-Brenes proposed Feature-Aware Student Knowledge Tracing (FAST)[10] to incorporate different skill features, such as subskills, problem's difficulty and student ability as parameters in KT model. Each parameter could be expressed in logistic regression for modeling the guessing and slipping probability with the skill features. Similarly, Yanbo[11] restructures the classic KT model using logistic regression over each step's subskills to model the transition probability including learning and forgetting for overall knowledge required by the step. All these feature-based extensions of BKT must rely upon experts to predefine the skill and subskill features without providing any automatic Q-matrix discovery method.

2.3 Item response theory

Item Response Theory (IRT) is a framework for modeling student responses on a set of assessments. The model is used to describe the relationship between the proficiency of a student and the likelihood of correctly answering a test item. It assumes that student proficiency is invariable in a test, and all the test items are measuring the same potential trait. In the two-parameter IRT model, the probability of the student s correctly answering the question q is given by:

$$p_{sq} = f(\alpha_q(\theta_s - \beta_q)) \quad (2)$$

We selected $f(x) = \Phi(x)$, where $\Phi(x)$ is the cumulative distribution function of the standard normal distribution. This model is known as the two-parameter ogive model.

Despite of the recent progress in the research of cognitive modeling for ITS, most modeling frameworks haven't presented an integrated and automatic solution that can tackle with the longitude student data and accurately describe the temporal development of individual students. **TABLE I** summarizes the status of the major proposals in the research community.

The major objective of the ATC framework is to present a unified modelling framework to address the above issue. From **TABLE I**, one can see the framework incorporates multi-dimensional and dynamic skill vector as well as the potential forgetting factor in the extended IRT system to describe the temporal learning sequence from student interaction events with educational contents. Moreover, the ATC employs the CFM model and Bayesian optimization to quickly identify the dimension range of latent skills, and perform its own IRT based statistical inference to calculate the optimal skill vectors.

TABLE I Comparison among CFM, KT, CDM and ATC framework

	Multi-Dimension Skill	Temporal Skill Development	Modeling Forgetting Factor	Automatic Q-Matrix Discovery and Refinement
CFM	Yes	Yes	No	No
Knowledge Tracing	Yes	Yes	Yes	No
CDM	Yes	No	No	Yes, Statistical Inference
ATC with IRT	Yes	Yes	Yes	Yes, Dimension Discovery + Statistical Inference

3. Automatic Temporal Cognitive Framework

The ATC framework presents a temporal probabilistic model describes dynamic student multi-dimensional skill level and the skills required for educational contents. Eq(3-4) represent the IRT function based skill embedding that projects a vector of multiple-dimensional latent skills into the a specific exercise:

$$q_{sit} = \frac{\vec{\theta}_{st} \cdot \vec{a}_i}{\|\vec{a}_i\|} - \|\vec{a}_i\| \quad (3)$$

$$p_{sit} = Pr(R_{sit} = 1 | \vec{\theta}_{st}, \vec{l}_i, \vec{a}_i) = \phi(q_{sit}) \quad (4)$$

Where

- ϕ is the logistic function convert a value to a probability between zero and one.
- $\vec{\theta}_{st}$ is the vector which represents the ability of each skill of student s at the timestep t .
- \vec{a}_i is the vector which represents the required skill level of exercise i .
- \vec{l}_i is the vector that represents the ability augmenter that a student can obtain after finishing question or education content i .
- R_{sit} is the result of the response of student s on question i of the timestep t .
- p_{sit} is the probability of the student s has a correct response on exercise i at timestep t .

According to Eq(3) and (4), the probability that the student s gives a correct response to the exercise i depends upon his ability $\vec{\theta}$ and the exercise's required skill level \vec{a} . We selected an exercise which requires two skills to plot the relationship between the student ability vector ($\vec{\theta}_s$) and the probability of correct response (p_{sit}). Figure 1 shows that the probability of correct response increases along with an increase in each element of the student ability vector. The figure also shows that the probability increases more quickly with changes in location parallel to the skill2-axis than changes parallel to the skill1-axis. The different increase rate corresponds to the difference in the \vec{a}_i parameters.

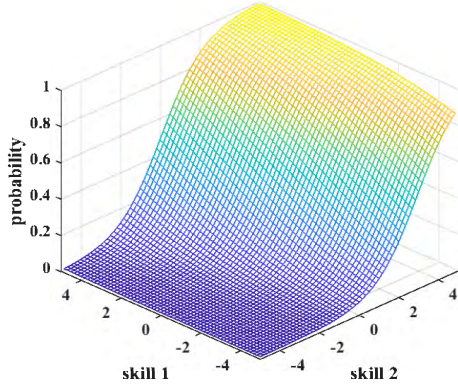


Figure 1 Probability of correct response for an exercise with given \vec{a}_i ($a_1 = 0.447$, $a_2 = 1.459$)

Eq(5-7) define a nonlinear dynamic system for hidden learning states of every student. They capture the temporal change between consecutive learning states and the skill improvement factor as well as the forgetting factor:

$$\theta_{s(t+1),n} \sim N(\mu_{s(t+1),n}, \sigma^2) \quad (5)$$

$$\mu_{s(t+1),n} = (\theta_{st,n} + l_{i,n} * \phi(q_{sit})) * f_{st,n} \quad (6)$$

$$f_{st,n} = \exp \left\{ - \left[\frac{1}{1 + \theta_{st,n}} * r + \beta \right] * \Delta t \right\} \quad (7)$$

- $\theta_{st,n}$ means the able of the n -th skill in the dimension of $\vec{\theta}_{st}$.
- $l_{i,n}$ represents the value of the n -th dimension of the vector \vec{l}_i .
- r and β are two parameters to fit.
- $f_{st,n}$ is the forget coefficient of student s from timestep t to timestep $t+1$.
- Δt is the interval between timestep t and timestep $t+1$.

Eq(5) assumes that latent skill $\theta_{s(t+1),n}$ at the moment $t+1$ follows the gaussian distribution with the mean $\mu_{s(t+1),n}$ depending upon the nonlinear transformation of $\theta_{st,n}$ on the previous moment. Eq(6-7) defines the transformation function that describes both knowledge acquisition and exponential forgetting factor in student learning process over the problem-answering sequence. Eq(7) indicates the forget coefficient is primarily determined by the current skill level of the student s and the time interval between two adjacent problem-answering events.

We estimated the model parameters for the ATC model by maximizing the following objective function:

$$L(\omega) = \text{Log Likelihood} = \sum_s \sum_{T_s} \log P(R | \vec{\theta}_{st}, \vec{l}_i, \vec{a}_i, \vec{f}_{st}) \quad (8)$$

Where ω is the model parameters, \mathcal{S} is the collection of all the students, T_s is the collection of all the answer timesteps of student s . We use Stan[12] to quickly implement a prototype of the ATC model. Stan is a state-of-the-art platform for statistical modeling and high-performance statistical computation. It's easy to specify log density functions in Stan's probabilistic programming language. Given the nonlinear nature of the ATC model, it is impossible to utilize EM method which is commonly used for linear space state models. Therefore, we employed HMC (Hamiltonian Monte Carlo) sampling in training the parameters of the ATC model according to the maximum likelihood goal.

4. Experiments

In our experiments, we choose area under the ROC curve (AUC) as the performance metric to measure the discriminative ability of the student model. Our experiments compared the following models:

- Model A: Non-negative Matrix Factorization and Conjunctive Factor Model (CFM). We combine both models to predict the probability of correct answer.
- Model B: ATC model without temporal skill improvement and forgetting factor.
- Model C: ATC model without the forgetting factor.

The experiments include two public datasets[13] and one simulated dataset. The public dataset was collected from Open Learning Initiative online course of biology from 2012 to 2014 include 5186 students and 4831 unique steps. We select chapter Cells and Gene from the entire dataset.

- 1) Dataset Cells: selected from chapter Cells and Chromosomes includes 68 students and 34 questions.
- 2) Dataset Gene: selected from chapter Gene Expression includes 99 students and 27 questions.
- 3) Simulated dataset: simulated data generated by model B

TABLE II Prediction performance of ATC and other models

Models	Dataset Cells		Dataset Gene	
	AUC	Loglikelihood	AUC	Loglikelihood
Model A	0.830	-347.664	0.662	-335.666
Model B	0.742	-1594.142	0.270	-1162.372
Model C	0.753	-9199.437	0.867	-4897.249
ATC model	0.872	-2902.272	0.793	-2322.128

4.1 Comparison of different models

The experimental results on the public datasets are summarized in TABLE II. It shows that the ATC model achieves the best AUC performance and its AUC is 17% higher than that of Model B in the dataset Cell. Model C achieves the best AUC and its AUC is 18.8% higher than that of Model B in the dataset Gene. It should be noted that the Model C without forgetting factor demonstrates better performance than the ATC model in the dataset Gene. We compared the total interval of two dataset and found that the datasets Gene has a shorter interval in the problem-answering sequence. The difference in the studying interval may contribute to the strength of the impact of the forgetting factor, which could explain why the ATC has slightly worse performance than Model C. Moreover, the Model A with the combination of Non-negative Matrix Factorization and CFM has worse performance than our ATC model in both datasets.

4.2 Comparison in different amount of data

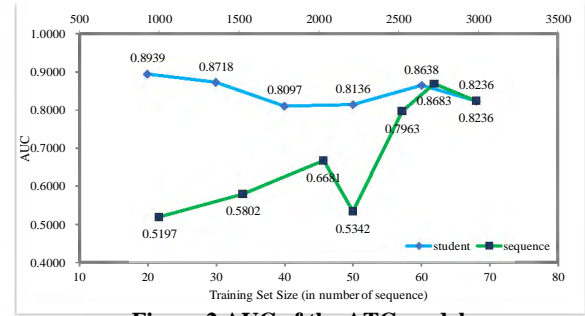
We compare the performance of our model in different data sizes with a changing sequence length and student number with the ATC model. Based on the dataset Cell, we run two groups of experiments: (1) In the first group of experiments, we change the sequence length of every student and keep the constant number of students. We choose seven datasets with the data size ranging from 800 to 2800 (under the same student number). (2) In the second group of experiments, we increase the number of student and keep the sequence length unchanged. We choose six datasets with the student number ranging from 20 to 70.

Figure 2 plots both experimental results. One can see that the AUC performance has an increasing tendency with the increase in the sequence length. It usually means that a longer problem-answering sequence can demonstrate dynamic of students' skill in a more accurate way. Furthermore, the increase in student number may slightly affect the performance of the ATC model and in some cases, it may even lead to degrade in the AUC performance.

These findings suggest further investigation should be made about the robustness of the ATC model on different lengths of problem-solving sequences and student number. Note that both factors can increase the number of parameters in the ATC model to be trained. Thus, more training experiments are needed to avoid possible overfitting or underfitting for the increase in complexity of the ATC model. Currently, the HMC process of training the ATC model is slow due to the large number of the model parameters. We are studying more efficient model training method for the ATC model to make sure that the model can be comprehensively evaluated in different datasets.

CONCLUSION

In this paper, we propose an Automatic Temporal Cognitive model

**Figure 2 AUC of the ATC model**

to trace student dynamic multidimensional ability and determine the related skill sets for educational contents in personalized learning system. This model regards the cognitive Q-Matrix as part of the model parameters and introduces nonlinear transmission functions to define student skill augmentation and forgetting factor. Experimental results confirm that our ATC approach is better than the traditional methods such as the combination of Non-negative Matrix Factorization and CFM. Moreover, the ATC model with skill improvement and forgetting factors tend to outperform the same multidimensional baseline model without these new features.

5. ACKNOWLEDGEMENTS

This work is supported by the Development Program of China (2018YFB1004502) and the NSFC (Grant No. 61532004).

1. REFERENCE

- [1] J. C. Stamper and K. R. Koedinger, "Human-machine student model discovery and improvement using DataShop," in *International Conference on Artificial Intelligence in Education*, 2011: Springer, pp. 353-360.
- [2] T. Barnes, "The Q-matrix method: Mining student response data for knowledge," in *American Association for Artificial Intelligence 2005 Educational Data Mining Workshop*, 2005, pp. 1-8.
- [3] T. Winters, C. Shelton, T. Payne, and G. Mei, "Topic extraction from item-level grades," in *American Association for Artificial Intelligence 2005 workshop on educational datamining*, Pittsburgh, PA, 2005, vol. 1, no. 2, p. 3.
- [4] J. M. Schraagen, S. F. Chipman, and V. L. Shalin, *Cognitive task analysis*. Psychology Press, 2000.
- [5] W. J. van der Linden and R. K. Hambleton, *Handbook of modern item response theory*. Springer Science & Business Media, 2013.
- [6] A. T. Corbett, J. R. J. U. m. Anderson, and u.-a. interaction, "Knowledge tracing: Modeling the acquisition of procedural knowledge," vol. 4, no. 4, pp. 253-278, 1994.
- [7] H. Cen, K. Koedinger, and B. Junker, "Comparing two IRT models for conjunctive skills," in *International Conference on Intelligent Tutoring Systems*, 2008: Springer, pp. 796-798.
- [8] N. Matsuda, T. Furukawa, N. Bier, and C. J. I. E. D. M. S. Faloutsos, "Machine Beats Experts: Automatic Discovery of Skill Models for Data-Driven Online Course Refinement," 2015.
- [9] J. P. González-Brenes and J. J. I. E. D. M. S. Mostow, "Dynamic Cognitive Tracing: Towards Unified Discovery of Student and Cognitive Models," 2012.
- [10] J. González-Brenes, Y. Huang, and P. Brusilovsky, "General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge," in *The 7th International Conference on Educational Data Mining*, 2014: University of Pittsburgh, pp. 84-91.
- [11] Y. Xu and J. Mostow, "Using Logistic Regression to Trace Multiple Sub-skills in a Dynamic Bayes Net," in *EDM*, 2011: Citeseer, pp. 241-246.
- [12] <https://mc-stan.org/>.
- [13] <https://pslcdatashop.web.cmu.edu/>.

Toward Instrumenting Makerspaces: Using Motion Sensors to Capture Students' Affective States in Open-Ended Learning Environments

Lucia Ramirez
Tufts University
Lucia.Ramirez@tufts.edu

William Yao
Harvard University
William_Yao@college.harvard.edu

Edwin Chng
Harvard University
Chng_weimingedwin@g.harvard.edu

Iulian Radu
Harvard University
Iulian_Radu@g.harvard.edu

Bertrand Schneider
Harvard University
Bertrand_Schneider@gse.harvard.edu

ABSTRACT

This paper investigates the potential of using an automated system for capturing students' body postures, location, and gestures in a makerspace as a means to quantify their affective states. In this study, Kinect sensors captured students' behavior in a fabrication lab over period of a 13-weeks semester, recording nearly half a million observations from 16 students enrolled in a class. Weekly surveys were conducted to serve as a ground truth for the students' affective states. Results from the survey were then analyzed in conjunction with the Kinect data to validate the setup of an automated system. Our preliminary findings suggest that our multi-sensor system is able to make a first step toward estimating students' levels of challenge, engagement and frustration. We conclude by describing how this kind of automated system provides a promising methodology for instrumenting makerspaces and supporting student learning in open-ended learning environments.

Keywords

Makerspaces, Open-ended Learning Environment, Instrumentation, Learning Analytics, Monitoring Tools

1. INTRODUCTION

Over the last decade makerspaces have garnered a great deal of attention given their potential for student's success in STEAM (Science, Technology, Engineering, Arts, and Mathematics) education [5]. The open environment, digital fabrication tools, and the community aspect of these spaces result in cultivating a growth mindset and creativity, which are often more effective than traditional learning experiences. Makerspaces typically involve person-to-person interactions and discussions throughout the development of an idea, design, and implementation, along with learning about modern tools such as 3D printers. More important from an educational perspective, those spaces support the development of important 21st skills such as problem solving and critical thinking.

A challenge within these largely open-ended learning environments, however, is the ability to measure those skills

Lucia Ramirez, William Yao, Edwin Chng, Iulian Radu and Bertrand Schneider "Toward Instrumenting Makerspaces: Using Motion Sensors to Capture Students' Affective States in Open-Ended Learning Environments" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 639 - 642

quantitatively and assess the effectiveness of pedagogical practices. A nascent field of research, Multimodal Learning Analytics (MMLA; [3]), proposes to use multimodal sensors to capture fine-grained information about students' learning trajectories. The ability to capture and track student behaviors can: 1) lead to a better understanding of the development of these skills; 2) inform instructors and facilitators of student's behavioral performance; 3) provide students with formative assessments; and 4) overall, discover new ways of supporting students' learning in open-ended learning environments.

2. RELATED WORK

In this section, we provide a rationale for studying the affective states of challenge, engagement and frustration from the perspective of Flow Theory and review related work on Multimodal Learning Analytics.

2.1 Flow Theory

Csikszentmihalyi (1990)'s flow theory [6] matches skill level of students and given challenge of tasks in order to reveal the affective states of students during their learning.

Since instructors would naturally avoid low challenge tasks that will not add value to students' learning, the concerned affective states for our study would only include flow and frustration. Flow is a multi-faceted construct that includes factors such as high levels of student engagement and distortions in the experience of time. The detection of engagement and frustration could be accomplished relatively straightforwardly using sensors that read body language and facial expression, but the recognition of flow state by sensors require further work that goes beyond the intended scope of this study. Therefore, for the purpose of this study, student engagement is employed as a crude approximation for flow, and student frustration is studied to inform instances when the challenge of the assigned tasks reaches above the skill level of the students.

2.2 Multimodal Learning Analytics

Over the last decade, the rise of the Maker Movement has motivated much research towards understanding the effectiveness and benefits of incorporating "making" activities in education. As research within these complex learning spaces continues, the approach of data collection is shifting from manual observation to more automated methods enabled by technological advancements and capabilities such as multimodal sensing capabilities.

The field of Multimodal Learning Analytics [3] has gained popularity over the last decade as a way to identify meaningful

metrics (or multimodal cues; [4]), that can be leveraged to deepen our understandings of students' states in such environments. Currently, most of MMLA research has been applied within a computerized learning context, controlled experiments (i.e., Randomized Control Trials; [1]) or formal learning environments (i.e., classrooms). For example, in a hands-on learning activity, Schneider & Blikstein (2015) used unsupervised machine learning algorithms on Kinect data and identified prototypical body postures indicating cycles of cognition and action - which were positively correlated with participants learning gains [9]. Using non-invasive methods, such as the Kinect sensor, to track a person's motion allows for practical implementation of assessing individuals' affective states, while preserving a natural environment.

Given that non-verbal cues such as body postures hold important information about mental states [8], this could allow motion tracking sensors to analyze students' affective states, such as perceived challenge of the tasks, levels of engagement and frustration. To our knowledge, the fusion of MMLA and the makerspace development is currently underexplored.

3. OVERVIEW

3.1 Course Curriculum

Over the course of fifteen weeks, the research team collected motion sensor data and survey responses of sixteen graduate students enrolled in a hands-on digital fabrication course. The course aimed to teach students the usage of modern fabrication technologies and their application to educational contexts. Students were responsible for implementing prototyping designs and building educational toolkits with the use these fabrication tools and associated software. Students were tasked with weekly assignments, either independently or in pairs, and asked to record their experiences after each assignment in a weekly survey.

3.2 Makerspace Setup

The makerspace was equipped with two Kinect v2 sensors to capture human motion within the space. The sensors were placed on opposing ends of the makerspace lab. Both sensors were connected to a Django server, where the Kinect data was stored.

3.3 Research Questions

- **RQ1:** At the group level, does this system accurately measure the levels of challenge, engagement and frustration of the class in the makerspace?
- **RQ2:** Can we detect individuals' levels of challenge, engagement, and frustration using the Kinect data?

4. METHODS

4.1 Survey Data

The survey data provides a ground truth to our analysis. In defining the survey, we leveraged prior studies conducted on interactions and learning in makerspaces, as well as surveys used by schools, universities and public libraries to assess the impact of making on students' learning. Making in fabrication labs has been described as offering "visceral design experiences", allowing for unprecedented levels of frustration, engagement and excitement experienced by students [2]. Thus, understanding participants' level of frustration, challenge, or even enjoyment of making assignments is key to validating such claims.

4.2 Kinect Data

We used two Kinect sensors to collect motion and posture data in the makerspace. Motion detection and tracking was possible by the embedded IR sensor within each Kinect sensors. For this study, the collected Kinect data included upper limb joints and binary values (e.g. left hand raised).

Cleaning and Labeling Kinect Data: In order to explore levels of frustration and engagement for an individual (RQ2), we needed to label the bodies in the room with the corresponding student. We collected face images generated by the Kinect sensors to solve the issue of person re-identification. To accomplish this labeling, we used an open-source face recognition algorithm: OpenFace, which produced an accuracy of 88% on average.

Standardizing the Kinect Data: The first step of pre-processing the joint data was to translate all 3-dimensional coordinate points to a reference system. Further data cleaning was facilitated using a video generation script that rendered videos of the makerspace next to a top-down mapping of the makerspace (Fig. 1).

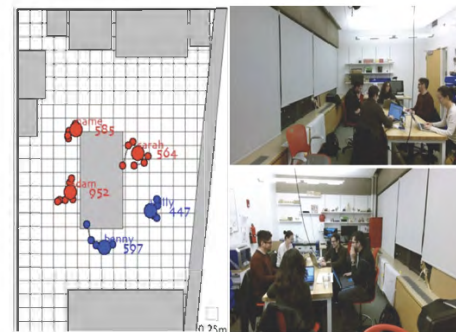


Figure 1. View of Kinect skeletons in space

Deduplicating the Kinect Data: Further data pre-processing consisted of deduplicating skeletons referencing the same person in the makerspace. Detecting duplicates was accomplished by finding head joint overlap within a given timeframe. The duplicate skeletons were then averaged or discarded given the similarity of the skeletons and location in respect to the Kinect sensors.

Generating features from the Kinect Data: Building on prior work [1; 7], we generated features from the Kinect data to capture students' levels of challenge, engagement and frustration. Based on the relative positioning of each skeletal joint, it is possible to generate meaningful features that can provide us with a basic understanding of the participants' body language (see Table 1):

Table 1. A few exemplar features generated from Kinect Data

Features	No. of features	Description
Joint Movement	7	Average of joint displacement between timeframes
Joint Angles	4	Average of joint angle
Lean	2	Average of body lean. Lean Left/Right or toward (forward) / away (back) from Kinect #1
Self - Touch	8	Average proximity between different joints vs. head; and joints vs. hand

In summary, the overall approach involved 1) collecting and cleaning the Kinect data, 2) generating features from skeletal joints,

3) detecting body language using features, and 4) mapping affective states of students.

5. RESULTS

RQ1: Can we detect levels of challenge, engagement and frustration of the class?

Fig. 2 shows a summary of the survey responses over all weeks, which indicates a parallel trend between levels of challenge and frustration, while levels of engagement consistently over a rating of 4. Because there seems to be a ceiling effect on the engagement measure, we focus on the detecting levels of frustration and challenge. Levels of challenge and frustration were very similar to each other (Fig. 2), which suggests that students experienced both at the same time. After generating correlation matrices between the survey and Kinect data, our results show that levels of challenge and frustration are highly correlated with body leaning and certain self-touch metrics (Fig. 3). On the other hand, aggregate levels of engagement did not show significant correlations with any of the generated Kinect metrics.

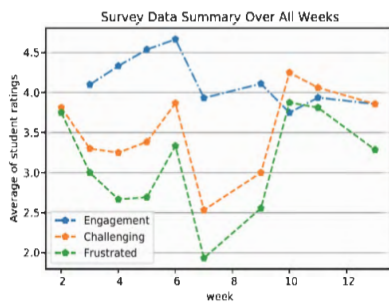


Figure 2. Survey summary of aggregate levels of engagement, challenge, and frustration

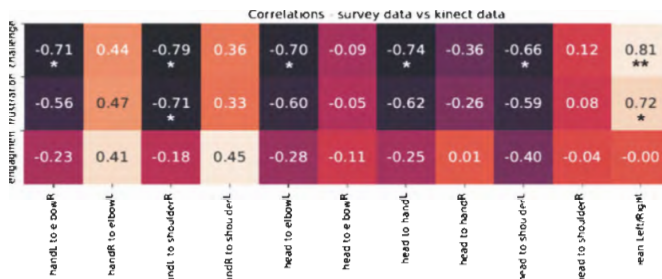


Figure 3. Survey and Kinect data showing that levels of engagement, challenge, and frustration are significantly correlated with self-touch metrics * $p \leq 0.01$; ** $p \leq 0.001$.

In order to further determine the key Kinect features that are associated with levels of challenge, engagement and frustration, we used decision trees to find the optimal combination of predictors. We focus on students' levels of frustration, since it overlaps with levels of challenge, and engagement suffered from a ceiling effect. We split the dataset into four quartiles and use a decision tree classifier to predict if the class was not frustrated (1st quartile), somewhat frustrated (2nd quartile), mildly frustrated (3rd quartile) or highly frustrated (4th quartile). Fig. 4 shows the resulting decision tree:

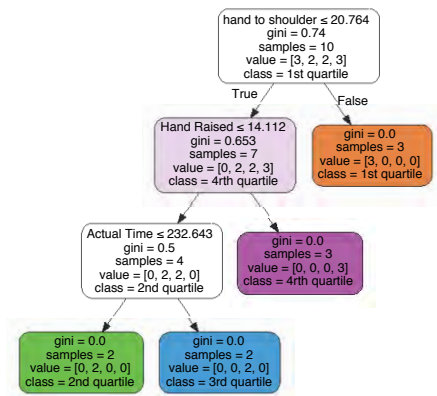


Figure 4: A decision tree used to predict levels of frustration.

Not surprisingly, the classifier is able to perfectly predict levels of frustration for each week of the semester. What interests us, however, is which combination of features is used to make this prediction. As found above, “hand to shoulder” was able to separate the 1st quartile (no frustration) from the other quartiles. “Hand raised” separated the 4th quartile (high frustration) from the other quartiles. Finally, the time spent in the makerspace separated the 2nd and the 3rd quartiles (somewhat / mildly frustrated).

RQ2: Can we detect levels of challenge, engagement and frustration of individual students?

The quantification of this metric is similar to that of RQ1, however, rather than aggregate measures, individual measures were analyzed after we identified Kinect data using a face recognition algorithm. Table 3 describes significant correlations between survey responses and Kinect metrics (defined in Table 1) for each individual. Table 3 shows that students express levels of engagement, frustration and challenge differently, and Table 2 summarizes the patterns found in Table 3. While some of these indicators are overall positively correlated with our dependent measures (e.g., hand movement / agitation, time spent in the makerspace) or negatively correlated with them (e.g., head to elbow, head to shoulder, head to hand), other indicators can be negatively correlated for some students and positively correlated for others (for example, shoulder angle is positively associated with frustration for Zoe and negatively correlated with frustration for Ann).

Table 2. Summary of correlation patterns found between individuals

	Positive	Negative
Challenge	Total time spent in the space, elbow and shoulder agitation, hand-to-elbow pose, and body leaning	Shoulder angle, head-to-elbow, head-to-shoulder, and head-to-hand
Frustration	Raising the hand	Head-to-elbow and elbow angle
Engagement	Hand agitation and raising the hand	Body leaning

Table 3. Cells show significant correlations for levels of frustration (F), challenge (C) and engagement (E) for every student. Orange: Students who expressed mid to high levels in F & C, but shared low levels of E; Red: Students who expressed high levels in all F, C & E.

	Dan	Ken	Sue	Bea	Ron	Zoe	Ann	Amy	Mia
actual time			0.87C			0.76F 0.79C			
agitation elbow, shoulder			0.72C	0.98E		0.88F 0.75C			
agitation hand			0.74C 0.96E	0.96E		0.89F 0.80C			
agitation head				0.99E		0.91F 0.77C			
angle elbow		-0.77E				0.77F 0.72C	-0.74F -0.81C		
angle shoulder		-0.77E				0.77F 0.72C	-0.78F -0.62C	-0.72C	
angle head		0.75F 0.73C -0.91E							
hand-to-elbow						-0.75C	0.69C -0.85E		
hand-to-shoulder	0.70E						-0.71F		-0.78F
head-to-elbow		0.78E			-0.66C		-0.78F -0.78C	-0.75C	0.74C
head-to-shoulder					-0.78C		-0.70F -0.68C	-0.81C	0.75C
head-to-hand	-0.64C			-0.89C			-0.77F -0.76C		-0.71E
lean left/right	0.74C				-0.66F -0.67C	0.87F 0.88C			
lean forward/back		-0.85E				-0.91E			
Raised hand					-0.71E	0.87F 0.71C			0.93E

6. DISCUSSION

Our data indicates that the use of motion sensors can potentially capture affective states in open-ended learning environments. Our findings provide some initial evidence that high correlation between levels of challenges and frustration is a signal that the course assignments may be above the skill levels of students; and frustration of the entire class can be approximated using self-touch gestures. Additionally, levels of engagement were high throughout the semester, which is not surprising given the nature of the class. Thus, it was difficult to find indicators of engagement from the Kinect data. Furthermore, these indicators vary across students. In sum, the correlation between the survey measures and the Kinect data suggest that this multi-sensor system can detect key affective states to inform students' learning trajectories within an open-environment such as the makerspace.

Our preliminary findings suggest that our multi-sensor system is able to make a first step toward estimating students' levels of frustration via affective states. Another contribution of this paper is to highlight the role of individual differences when using this kind of monitoring tools. For a multimodal sensor system to be accurate and successful in supporting teachers and students, it has to take into account these individual differences in ways of experiencing the space.

7. CONCLUSION

Because makerspaces have gained so much popularity over the last decade, it is becoming increasingly important to design rigorous ways of studying them. The multi-sensor system described in this paper makes a first step toward an effective semi-automated methodology for instrumenting makerspaces. While these findings rely on simple measures, they pave the way for more complex data collection and data analysis techniques.

The implementation of such multi-sensor systems in open-ended learning spaces has the potential to 1) attain a better understanding of pertinent skill development; 2) provide students with formative assessments; 3) inform instructors and facilitators of student's behavioral performance; and 4) discover new ways of supporting a student's learning overall. As our understanding of a classroom is constantly being redefined, Multimodal Learning Analytics [3] can certainly assist with optimizing the use of these evolving and varied learning environments.

8. REFERENCES

- [1] Behoora, I., & Tucker, C. S. (2014, August). Quantifying emotional states based on body language data using noninvasive sensors. In ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (pp. V01AT02A079-V01AT02A079). American Society of Mechanical Engineers.
- [2] Blikstein, P. (2013). Digital Fabrication and 'Making' in Education: The Democratization of Invention. In J. WalterHerrmann & C. Büching (Eds.), *FabLabs: Of Machines, Makers and Inventors*. Bielefeld: Transcript Publishers.
- [3] Blikstein, P., & Worsley, M. (2016). Multimodal Learning Analytics and Education Data Mining: Using computational technologies to measure complex learning tasks. *Journal of Learning Analytics*, 3(2), 220–238.
- [4] Chen, L., Feng, G., Joe, J., Leong, C. W., Kitchen, C., & Lee, C. M. (2014, November). Towards automated assessment of public speaking skills using multimodal cues. In Proceedings of the 16th International Conference on Multimodal Interaction (pp. 200-203). ACM.
- [5] Clapp, E. P., Ross, J., Ryan, J. O., & Tishman, S. (2016). *Maker-centered learning: Empowering young people to shape their worlds*. San Francisco, CA: Jossey-Bass.
- [6] Csikszentmihalyi, M. (1990) *Flow: The psychology of optimal experience*. New York: Harper & Row.
- [7] Guntz, T., Balzarini, R., Vaufreydaz, D., & Crowley, J. (2018). Multimodal Observation and Classification of People Engaged in Problem Solving: Application to Chess Players. *Multimodal Technologies and Interaction*, 2(2), 11.
- [8] Phutela, D. (2015). The importance of non-verbal communication. *IUP Journal of Soft Skills*, 9(4), 43.
- [9] Schneider, B., & Blikstein, P. (2015). Unraveling Students' Interaction Around a Tangible Interface using Multimodal Learning Analytics. *International Journal of Educational Data Mining*, 7(3), 89-116

Exploring Stealth Assessment via Deep Learning in an Open-Ended Virtual Environment

Joseph M. Reilly

Harvard Graduate School of Education

13 Appian Way

Cambridge, MA 02138

1 (617) 496-5164

josephreilly@fas.harvard.edu

Chris Dede

Harvard Graduate School of Education

13 Appian Way

Cambridge, MA 02138

1 (617) 495-3839

chris_dede@gse.harvard.edu

ABSTRACT

Rich streams of fine-grained activity data are generated by many game-based learning environments. These data can be mined unobtrusively to assess student learning and potentially foster tailored feedback and scaffolding. Stealth assessment frameworks are more commonly seen in constrained virtual environments with clear goals and markers of progress; these diagnostic/formative-feedback strategies are more difficult to implement in open-ended environments with a larger range of potential activities and more variation in problem-solving strategies. This study investigates the potential for using a long short-term memory network (LSTM)-based model to enable stealth assessment in an immersive virtual world for learning. We investigate how well these temporally-sensitive models can predict learning gains on a pre-post survey, as well as predicting the quality of a student-generated concept map. This method is applied to log file data from a middle school science curriculum based on an open-ended immersive virtual environment. Results indicate that the LSTM-based model outperforms support vector machine and random forest models and can accurately predict performance without knowing pre-scores. This deep learning approach is promising for stealth assessment and formative feedback in open-ended virtual environments.

Keywords

Stealth assessment, immersive virtual world, deep learning, recurrent neural network, LSTM.

1. INTRODUCTION

Technology-enhanced learning opportunities through game-based learning and assessment are becoming more common in classrooms, and these novel forms of teaching require a rethinking of how to assess student learning across digital and physical spaces [1]. With an increased research focus on how game-based learning can promote 21st century skills [2], scholars and developers of virtual environments are more frequently considering how difficult-to-measure constructs like problem solving and scientific reasoning can be taught and evaluated. Structured simulations have been used for decades in science to model and explore difficult to perceive or

manipulate phenomena [3,4] but these tools are typically isolated as specific laboratory experiments and are not embedded in authentic frameworks. More complex and less structured environments designed for inquiry learning offer chances for social interaction and participation in the practices of science [5].

Unlike traditional summative assessment of student work, stealth assessment can be employed in virtual settings to diagnostically gauge student competency across a variety of constructs without interrupting immersion and learning [6]. Within the framework of evidence-centered design, virtual tasks are designed to generate logged actions that can act as evidence for a competency model of a certain construct [7]. This type of formative assessment has been applied to a variety of epistemic games as well as games for learning [8] and can provide a foundation for developing feedback mechanisms and dynamic scaffolding to students [9]. This approach to assessment and feedback generation encounters challenges when applied to open-ended virtual environments.

This study presents several ways of using recurrent neural networks trained on student logged actions over time in an open-ended virtual environment for learning in order to predict student learning across several constructs as well as the quality of a student-generated concept map. The effects of temporality in the data and the impact of including pre-survey measures are explored.

2. LITERATURE REVIEW

Well-designed educational games and computer-based simulations engage and excite learners while offering a chance to interact with a dynamic virtual world [10]. Virtual environments have been used effectively in science education to foster inquiry and learning not only of content, but also of the fundamental frameworks and practices of science [13]. Game-driven intelligent tutoring systems and game-based assessments have driven the field to consider the role games and assessments have in case-based teaching [14], how higher-order inquiry skills can be assessed in science [15], and how traditional psychometrical considerations do or do not apply to these new tools [16].

Stealth assessment and educational data mining of virtual worlds have been applied successfully to a variety of science-oriented games and curricula. In-game actions of students correlated significantly with pre-post content assessments [17]. Other work explored how tool use and completion of specific content generates multiple potential pathways students take in successfully completing tasks [9]. Determining implicit understanding of science concepts based on observed strategic moves in game-based assessments is also possible, although it still typically requires significant human coding [19].

Joseph Reilly and Chris Dede "Exploring Stealth Assessment via Deep Learning in an Open-Ended Virtual Environment" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 643 - 646

More researchers are exploring the applications of deep learning techniques to stealth assessment. Min et al. proposed the *DeepStealth* framework, which showed stacked denoising autoencoders could significantly outperform traditional machine learning methods in predicting student gains during a game-based learning curriculum [20]. LSTM-based models seem well-suited to the task of stealth assessment, outperforming comparable models in stealth assessment with no hand-engineering of features [21]. These models can also be combined with unsupervised learning techniques to identify problem-solving behaviors in learners and to predict learning gains [4].

Many of these techniques, however, do not work for more open-ended environments. Akram et al. note that their clustering approach, “is not appropriate for analyzing ill-defined problems where players are not bound to certain actions” [4]. In less strictly-defined learning environments, there is no set series of challenges to tackle or sub-goals to accomplish in service of a defined goal. Instead, users are free to revisit and repeat aspects of the world, do things in whatever order they see fit, and generally proceed in more erratic ways.

3. RESEARCH QUESTIONS

This study attempts to answer the following research questions:

1. Can team learning gains in the EcoXPT curriculum be accurately predicted from their logged trace actions?
2. What impact do the survey features have on the ability to predict learning gains?
3. How important is the temporality of data for the LSTM-based models?
4. Can the quality of teams’ final concept maps be accurately predicted from their logged trace actions?

4. ECOXPT CURRICULUM

4.1 Overview

The EcoXPT virtual world is the foundation of a twelve-day ecosystem science curriculum designed for use in middle school science classes [12]. Students work in small teams (2-3 students with one device) to explore a virtual pond ecosystem and its watershed, talk to non-player characters in the virtual world, collect data about water quality, and conduct authentic experiments. This inquiry-based curriculum teaches ecosystems science, scientific inquiry, and complex causality. There are no specific quests given or sub-tasks assigned, and no one “correct” answer is expected from teams. Many static scaffolds are built into the virtual world, but feedback is not based on the teams’ actions.

4.2 Dataset

The data for this study was collected as part of two larger studies comparing EcoXPT to an earlier immersive world-based curriculum developed by our team. After cleaning, logs from 320 teams of students from 16 different teachers were analyzed. All schools in this sample are urban/suburban public middle schools located in the northeastern United States. Logged records of student actions containing no identifiable information were stored during every session in a PostgreSQL database.

During this study, participating students completed a pre-post survey with six constructs: affective dimensions, ecosystem science content, understanding of causality, correlation versus causality,

experimental methods, and epistemology. As a final deliverable and representation of each team’s conclusive understanding of the complex causal web of interactions that lead to the fish dying, a causal concept map tool is used to show how factors in the ecosystem affect each other. Students drag nodes onto a board and connect the nodes with arrows indicating the direction of causation. Prior work has automatically analyzed student concept maps to evaluate overall structure, inclusion of correct and incorrect claims, and the types of evidence used by claim [11].

5. METHODS

Our main aim is to evaluate how well teams’ in-world actions can predict gains on our surveys as well as the quality of their concept maps. These logged actions contain meaningful temporal relationships, as these series of interactions represent problem-solving and exploratory strategies teams employed during the curriculum. Like other recent work on modeling temporal dependencies in log file data, we use a classifier model based on a LSTM. We also utilized more traditional classifiers that do not account for the temporal relationships in our data: a support vector machine (SVM) and a random forest (RF) model. In order to move beyond pre-post assessment and towards stealth assessment, several different versions of our features were used. One version contained only pre-test information, the second contained only logged actions, and a third feature set contained both. This allowed us to estimate the importance of the temporal dependencies the LSTM-based model leverages and showed how much more predictive power those models afford.

5.1 Data Pre-processing

This paper focuses on gains in three of the six total constructs: ecosystem science content, understanding causality, and experimental methods. Students completed the surveys as individuals, so average team pre- and post-scores were calculated for all teams in the study. As done in [4], survey performance was divided into low, medium, and high performers by setting thresholds at tertile values. This creates roughly balanced classes and allows simple interpretation of all survey constructs. Team concept maps were scored for accuracy according to a rubric designed by an ecosystem science expert [11] that separates claims made by students into core claims essential for understanding the complex causality of the ecosystem, tangentially related claims, and incorrect claims.

To reduce the dimensionality of the data, we binned our 68 events into six coarser-grained meta-categories that encompassed the range of different activity types available in the EcoXPT virtual world [18]: explore, collect, analyze, experiment, hypothesize, and tutorial. After classification, these categories were one-hot encoded. Combined with one-hot encoded tertile assignments for student pre-scores on the survey constructs, this resulted in nine features for use in classification. For the random forest and support vector machine models, cumulative counts of the one-hot encoded meta-categories were calculated. Each team thus had one observation of nine features. The LSTM model uses a three-dimensional tensor with dimensions of [samples, time steps, features].

5.2 Classifiers

We utilized the sci-kit learn package for Python to train random forest and SVM classifiers on our data and Keras for the LSTM-based model. Two-thirds of our teams were randomly selected to

act as a training set, while the remaining third of teams act as the test set. 5-fold cross-validation was used to tune hyperparameters.

As a baseline, the most common label for all classes for each survey or concept map was assigned to all teams. Grid search with 5-fold cross-validation was used to optimize the regularization parameter of the SVM model for each of our feature sets. As with the SVM model, grid search with 5-fold cross-validation was used to optimize the number of trees used.

Due to the lengthy training time of recurrent neural networks, LSTM hyperparameters were chosen initially based on those from similar literature [21, 4] then tuned by hand. Our final LSTM-based model was kept intentionally simple to reduce overfitting on a small sample of team logs. A single bidirectional long short-term memory layer with three neurons was used along with a dropout of 0.75 to both the input and recurrent connections. This layer output to a fully connected layer with a softmax activation that made the class assignments. The model utilized the Adam optimizer with a categorical cross-entropy loss function. The models could train for up to a maximum of 100 epochs.

6. RESULTS

6.1 Predicting Post Scores

The accuracies of our baseline classifier as well as our three classification models for three survey constructs are summarized in Table 1. The highest accuracy for each construct is in bold.

Table 1. Accuracy of classifiers when predicting post-score on three survey constructs.

Classifier	Content	Causality	Methods
Baseline	36.1%	35.1%	38.1%
SVM	63.2%	73.4%	69.4%
RF	61.2%	75.5%	68.4%
LSTM	68.8%	78.6%	73.5%

It is likely that this dataset’s use of pre-determined meta-classifications is more likely to yield higher accuracy for all models as this feature engineering step reduces dimensionality and imparts domain knowledge in the feature set. We attempted to use less-constrained feature sets as inputs for the LSTM-based models but issues of overfitting, training time, and limited memory capacity were limitations for the current analyses.

6.2 Importance of Survey Features

We investigated the effectiveness of using solely our logged user activities by constructing a partial feature set that omitted team pre-survey data. The performance of our LSTM-based model on both feature sets is shown in Table 2.

Table 2. Performance of LSTM-based model when predicting content post-score with and without pre-score information.

Features	Accuracy	Precision	Recall	F1 Score
Only logs	50.5%	0.255	0.505	0.339
Combined	68.8%	0.742	0.687	0.637

The inclusion of the pre-survey performance features improves model accuracy by roughly 17%. This indicates that a large amount of predictive power in the model comes from the logged features.

6.3 Predicting Concept Map Quality

Using the quality of student-generated concept maps as a target variable allows us to use solely data from the virtual world as both target and predictor variables. The performance of our three classifiers is reported in Table 3.

Table 3. Performance of classifiers when predicting concept map quality.

Features	Accuracy	Precision	Recall	F1 Score
SVM	37.9%	0.392	0.379	0.371
RF	43.9%	0.462	0.455	0.442
LSTM	48.5%	0.339	0.485	0.399

6.4 Importance of Temporality

In order to assess the importance of the order of logged actions for making accurate predictions, a new deep learning model was fit that did not have any recurrent layers and where connections do not form a cycle to learn from previous sequences. A feedforward neural network consisting of two fully connected layers with six hidden units each and “relu” activation was trained to accomplish the same concept map quality prediction task described in Section 6.3. This model’s accuracy was able to match that of the random forest (43.9%) but could not match that of the LSTM network.

7. DISCUSSION

The LSTM-based models consistently outperform their the highest performing SVM and RF models as well as the majority class baseline. These performance differences are like those reported in similar deep learning stealth assessment models [21]. While a small amount of feature engineering based on expert content knowledge is done here to reduce dimensionality of the feature set, automated methods of doing so in the future could remove even this barrier to fully automatic stealth assessment of open-world environments. These classifications can act as formative assessments for teachers to direct their attention during implementation of the authentic scientific inquiry curriculum to students who most need help. Additionally, fail-soft interventions could be included in the program itself that trigger based on patterns in the longitudinal record of teams’ actions.

Our small sample size might lead to low accuracy. Our current analysis also ignores the role of the class and teacher that teams are assigned to. Ongoing work in our group on teacher fidelity of implementation shows that teachers vary significantly in the amount and quality of support offered to their classes. Our current concept map scoring metric is a work-in-progress, and we may be able to triangulate our concept map scoring methods with observed learning gains to ensure that our dependent variable is an accurate representation of team understanding of causality.

The techniques presented here are a first step toward applying deep learning stealth assessment techniques to an open-world immersive virtual environment for learning. Future work will take a more fine-grained look at logged events with a focus on exploring how actions within the different experimental tools might afford a better assessment of student understanding of causality. Employing concept coding strategies via unsupervised methods to classify different types of maps may allow for more automation and increased generalizability of this method. Teams enter free response text as reasoning to justify their claims in the concept map and these data have not yet been analyzed. Natural language

processing techniques may lead to a deeper understanding of how teams use causal reasoning when making claims.

8. CONCLUSION

This study successfully applied deep learning techniques from games for learning literature to open-ended environments for learning. Using data collected from several implementations of a multi-week virtual environment-based curriculum, we evaluated several popular classifiers in the educational data mining literature and showed that a LSTM-based model consistently outperformed them. These models can operate unobtrusively while teams make their way through the curriculum. Classifications regarding the capabilities of teams across several different constructs can inform interventions administered directly through the virtual world or flag certain teams for the classroom teacher to follow up with. Future work will involve further optimization and regularizations of the model to improve performance as well as training the assessor on other versions of the logfile data with different grain sizes.

9. ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation through Grant 1416781 and the Cheng Yu Tung Research Innovation Fund. The opinions expressed are those of the authors and do not represent views of the National Science Foundation.

10. REFERENCES

- [1] Crisp, G.T., 2014. Assessment in next generation learning spaces. In *The Future of Learning and Teaching in Next Generation Learning Spaces*, 85-100. Emerald Group Publishing Limited.
- [2] Qian, M. and Clark, K.R., 2016. Game-based Learning and 21st century skills: A review of recent research. *Computers in Human Behavior*, 63, 50-58.
- [3] Wieman, C.E., Adams, W.K. and Perkins, K.K., 2008. PhET: Simulations that enhance learning. *Science*, 322, 682-683.
- [4] Akram, B., Mott, B., Min, W., Boyer, K.E., Wiebe, E. and Lester, J., 2018. Improving Stealth Assessment in Game-based Learning with LSTM-based Analytics. In K.E. Boyer & M. Yudelson (Eds.), *Proceedings of the 11th International Conference on Educational Data Mining*, 208 – 218.
- [5] Barab, S.A., Sadler, T.D., Heiselt, C., Hickey, D. and Zuiker, S., 2007. Relating narrative, inquiry, and inscriptions: Supporting consequential play. *Journal of science education and technology*, 16(1), 59-82.
- [6] Shute, V.J., 2011. Stealth assessment in computer-based games to support learning. *Computer Games and Instruction*, 55(2), 503-524.
- [7] Mislevy, R.J., Almond, R.G. and Lukas, J.F., 2003. A brief introduction to evidence-centered design. *ETS Research Report Series*, 1-29.
- [8] Rupp, A.A., Gushta, M., Mislevy, R.J. and Shaffer, D.W., 2010. Evidence-centered design of epistemic games: Measurement principles for complex learning environments. *The Journal of Technology, Learning and Assessment*, 8(4).
- [9] M. Cheng, L. Rosenheck, C. Lin, and E. Klopfer. 2017. Analyzing gameplay data to inform feedback loops in the Radix Endeavor. *Computers & Education*, 111, 60–73.
- [10] Squire, K., 2011. Video games and learning: Teaching and participatory culture in the digital age. New York, NY: Teachers College Print.
- [11] Reilly, J., Kamarainen, A., Metcalf, S., Dede, C. and Grotzer, T. 2019. *The Importance of Time and Sequence on Learning in Mobile Augmented Reality*. Paper presented at the 92nd Annual International Conference of NARST.
- [12] Dede, C., Grotzer, T.A., Kamarainen, A. and Metcalf, S., 2017. EcoXPT: Designing for deeper learning through experimentation in an immersive virtual ecosystem. *Journal of Educational Technology & Society*, 20(4), 166-178.
- [13] Nelson, B. C., Kim, Y., Foshee, C., & Slack, K. (2014). Visual signaling in virtual world-based assessments: The SAVE Science project. *Information Sciences*, 264, 32-40.
- [14] Gómez-Martín, M.A., Gómez-Martín, P.P. and González-Calero, P.A., 2004, September. Game-driven intelligent tutoring systems. In *International Conference on Entertainment Computing*, 108-113.
- [15] Clarke-Midura, J. and Dede, C. 2010. Assessment, technology, and change. *Journal of Research on Technology in Education*, 42(3), 309-328.
- [16] Mislevy, R.J., Oranje, A., Bauer, M.I., von Davier, A.A., Hao, J., Corrigan, S., Hoffman, E., DiCerbo, K. and John, M., 2014. Psychometric considerations in game-based assessment. In *Technology and Testing: Improving Educational and Psychological Measurement*.
- [17] Shute, V.J., Ventura, M. and Kim, Y.J., 2013. Assessment and learning of qualitative physics in newton's playground. *The Journal of Educational Research*, 106(6), 423-430.
- [18] Reilly, J. and Dede, C. 2019. Differences in Student Trajectories via Filtered Time Series Analysis in an Immersive Virtual World. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, 130-134.
- [19] E. Rowe, R. Baker, J. Asbell-Clarke, E. Kasman, and W. Hawkins. 2014. Building automated detectors of gameplay strategies to measure implicit science learning. In *Proceedings of the 7th International Conference on Educational Data Mining*, 337-338.
- [20] Min, W., Frankosky, M.H., Mott, B.W., Rowe, J.P., Wiebe, E., Boyer, K.E. and Lester, J.C., 2015, June. DeepStealth: leveraging deep learning models for stealth assessment in game-based learning environments. In *International Conference on Artificial Intelligence in Education*, 277-286.
- [21] Min, W., Frankosky, M.H., Mott, B.W., Wiebe, E.N., Boyer, K.E. and Lester, J.C., 2017, June. Inducing stealth assessors from game interaction data. In *International Conference on Artificial Intelligence in Education*, 212-223.

Balancing Student Success and Inferring Personalized Effects in Dynamic Experiments

Hammad Shaikh
University of Toronto
hammy.shaikh@mail.utoronto.ca

Arghavan Modiri
University of Toronto
modiri.arghavan@gmail.com

Joseph Jay Williams
University of Toronto
williams@cs.toronto.edu

Anna N. Rafferty
Carleton College
arafferty@carleton.edu

ABSTRACT

Randomized controlled trials (RCTs) can be embedded in educational technologies to evaluate how interventions affect student outcomes and how effectiveness varies with characteristics like prior knowledge. But RCTs often assign many students to ineffective conditions. Adaptive algorithms like contextual multi-armed bandits (MABs) could change how students are assigned to conditions over time, offering the potential to both evaluate effectiveness for subgroups of students and direct more students to interventions that are effective for them. We use simulations to compare contextual MABs to traditional RCTs and non-contextual MABs. Contextual MABs improve outcomes for each subgroup; in contrast, non-contextual MABs may help one group of students, such as those with high prior knowledge, while hurting another. Because both MAB algorithms adaptively assign conditions based on prior students' results, both recover biased estimates of condition effectiveness. However data collected from a contextual MAB is still nearly as good for inferring the optimal assignment policy as from an RCT.

1. INTRODUCTION

Randomized controlled trials (RCTs, sometimes known as A/B testing in software) can identify both how effective an intervention is overall and whether its effectiveness varies systematically based on individual characteristics of students (as in, e.g., [2]). Yet, RCTs may assign many students to ineffective conditions. Adaptive experiments can help by adjusting condition assignments based on their effectiveness for previous students. Multi-armed bandit (MAB) algorithms have been explored as a way of conducting such adaptive experiments in educational technologies [8, 11].

We explore the impact of MAB algorithms when there are participant-treatment interactions with different optimal con-

ditions for subgroups of learners. For instance, students with lower prior knowledge benefit more from concrete versus abstract examples, while the opposite holds for students with higher prior knowledge [2]. *Contextual* MABs have the potential to learn how effective conditions are for individuals [7]. These algorithms estimate condition effectiveness as a function of features, such as prior knowledge. We propose using contextual MABs to conduct personalized experiments that assign students to conditions based on their characteristics and simultaneously estimate how these characteristics impact the (differential) effectiveness of the conditions.

However, collecting data via contextual MABs has the potential to introduce biases: unlike in an RCT, condition assignment is dependent on an individual's characteristics and the assignments and outcomes of previous students. When conducting experiments using non-contextual MAB algorithms, the latter feature can bias the measured effectiveness of conditions, skewing the results of hypothesis tests [5, 9].

In this paper, we explore the tradeoff between *maximizing* how many students receive the most effective version of a technology for their individual learning needs and *estimating* the effectiveness of different versions of a technology. Using simulations, we compare three ways to assign students to experimental conditions, such as the different versions of an educational technology: uniform random assignment as in a typical RCT, a non-contextual MAB as in [11], and a contextual MAB that adaptively personalizes based on student characteristics. We consider scenarios in which a single feature, such as a student's prior knowledge, impacts what condition leads to the best outcome for the student, and examine cases where there is also an overall main effect of condition (i.e., averaged across all students, one condition is better than the other) versus cases where there is only an interaction effect between the student feature and the condition. Specifically, we investigate the following questions:

1. When does a contextual MAB assign more students to the most effective condition for them than the other policies, and how does this vary over the course of the experiment?
2. To what extent is there bias in the estimates of condition means for different subgroups? How does this impact the probability of inferring the optimal policy from the data?

We find that contextual MAB improves student outcomes compared to a typical RCT or non-contextual MAB. Fur-

Hammad Shaikh, Arghavan Modiri, Joseph Jay Williams and Anna Rafferty "Balancing Student Success and Inferring Personalized Effects in Dynamic Experiments" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 647 - 650

thermore the RCT is only slightly better than the contextual MAB in inferring the optimal policy. However both MABs introduce some bias: this manifests in the estimation of the participant-treatment interaction effect for contextual MABs, versus the main effect for non-contextual MABs.

2. CONTEXTUAL MAB ALGORITHMS

MAB algorithms are used in online decision-making scenarios in which a system must repeatedly choose an action in an effort to maximize reward. Initially, the relation between reward and actions is unknown to the system. A *contextual* MAB algorithm is used if an action's reward depends on both the action itself and the features of the data point (i.e., the context). Throughout this paper, we consider the following example of an educational experiment as a contextual MAB problem: students complete a homework activity, and are assigned to one of two experimental conditions: video or text hints. The experimental conditions are *actions*, and the *reward* is the student's score on the next homework activity. The effectiveness of text versus video hints is not known at the outset, and may vary according to the student's prior knowledge: *low* versus *high*. The system learns this relationship over time. More formally, for each student i with prior knowledge X_i , the system assigns a condition a_i and observes the resulting homework score R_i . To assign a condition to student $i + 1$, the system uses the student's prior knowledge X_{i+1} and the outcomes of all previous conditions assigned to students $1, \dots, i$. The optimal *policy* for assigning a student to a condition is the policy that chooses the action a^* with the highest expected reward based on the student's prior knowledge: $a_i^* = \operatorname{argmax}_a E[R_i(a) | X_i]$, where $R_i(a)$ refers to the reward (e.g., homework score) for student i as a function of the chosen action.

Contextual MAB algorithms must balance exploring what actions are effective based on the feature value(s) and exploiting information learned from the past action choices. Most contextual MAB algorithms, including Thompson Sampling [1] which is used in this paper, seek to minimize *cumulative average regret*: mean regret after n students is

$$\frac{1}{n} \sum_{i=1}^n E[R_i(a_i^*) | X_i] - E[R_i(a_i^{\text{actual}}) | X_i],$$

where a_i^{actual} is the actual action chosen for student i . In this paper, we focus on parametric contextual bandits where the expected reward $E[R_i(a) | X_i]$ is modeled as a linear regression function. In our example, assume we formulate the low or high prior knowledge of student i with $X_i = 1$ or $X_i = 0$, respectively. Then, we can use the following regression equation to describe the reward function:

$$E(Y_i | D_i, X_i) = \beta_0 + \beta_1 D_i + \beta_2 D_i X_i, \quad (1)$$

where D_i is 1 if student i is assigned to text hints and 0 for video hints. Note that we assume the contextual feature does not impact the outcome to simplify the exposition of our results. A primary benefit of this linear model is that each parameter represents a quantity of practical interest to researchers. For instance, $\beta_0 = E(Y_i | D_i = 0, X_i = 0)$ is the average achievement for students with high prior knowledge who received a video hint, and β_1 is the effect on achievement for text hints relative to video for students with high prior knowledge. Finally, β_2 is the differential benefit of re-

ceiving text hints for low prior knowledge students relative to students with high prior knowledge.

In this paper, we use Thompson sampling [1], which stochastically selects an action based on its probability of being the optimal action. Thompson sampling periodically estimates the regression equation (equation 1) using regularized Bayesian regression. We assume a conjugate normal-inverse gamma $N\text{-}\Gamma(0, \mathbf{I}, 0, 0)$; the prior corresponds to L_2 regularization (i.e., ridge regression), penalizing large coefficients.

3. RELATED WORK

MAB algorithms have been used in past work to conduct experiments within educational technologies, such as identifying the most effective explanations in an online quiz [11]. Contextual MAB algorithms have also been used in personalized intelligent tutoring systems in which the algorithm selects problems with appropriate difficulty level based on the students' profile [3]. In this paper we add to this literature by exploring the trade-off between assigning a higher proportion of students to their optimal intervention while also learning how the impact of an intervention varies across contextual features of students, similar to the exploration in [9] for non-contextual MABs.

Within the MAB and contextual MAB literature, identifying differences among actions (here, experimental conditions) and reliably measuring the degree of difference is an active area of research. In situations where user features are likely to have high impact on outcomes and there are many actions, randomized experiments can be impractical due to a combinatorial explosion of possibilities [6]. That work shows contextual MABs can be effective in high dimensions, motivating some of the current work. Several MAB algorithms have been proposed that change the loss function to increase measurement accuracy [5] or increase the chance of correctly identifying causal effects [10], and other work has developed MAB variants that correct for biases in estimated effects by re-weighting the data [4]. Our research builds on this prior work by detailing the extent of the measurement bias in a standard contextual MAB algorithm and investigating bias due to use of a non-contextual MAB when participant-treatment interactions are present, such as a student's prior knowledge influencing condition effectiveness.

4. INVESTIGATING MAB IMPACT

In simulations, we investigate the effectiveness of adaptive experiments using a MAB versus a standard RCT both in terms of average student outcomes and for evaluating the relative effectiveness of different conditions. When designing an adaptive experiment, the researcher must decide whether participant-treatment interactions should be included by deciding whether to use a contextual or non-contextual MAB algorithm. We explore the consequences of incorrectly using a non-contextual MAB in a situation where condition effectiveness is dependent on student's prior knowledge. We predict that contextual MABs will be most effective at assigning students to the best condition for them, and even when the non-contextual MAB recognizes that one condition is on average more effective than another, the contextual MAB will learn this more quickly. However, it's likely that both types of MAB algorithms may overestimate effect sizes, consistent with biases in experiments without contextual effects [9].

4.1 Simulation Framework

All simulations assume a sample of 1000 students. This is the size of a very large university classroom or the number of active users in an average MOOC; we also report on results after 250 students, corresponding to a somewhat smaller university class. We assume a binary condition $D_i = I(i \text{ in treatment group})$ and a single binary contextual variable X_i . The binary contextual variable of our study follows a Bernoulli distribution with $P(X_i = 1) = 0.5$. Such binary variables are common in educational contexts, such as when representing a student's prior knowledge as below or above the median student. Outcomes (reward) are generated using a linear regression model that estimates the reward of providing two groups of students (low versus high prior knowledge, differentiated by $X_i = 1$ versus $X_i = 0$) with a video or text hint ($D_i = 0$ or $D_i = 1$):

$$Y_i = \beta_0 + \beta_1 D_i + \beta_2 D_i \times X_i + \epsilon_i,$$

where $\beta = (\beta_0, \beta_1, \beta_2)$ are the parameters of interest and $\epsilon_i \sim N(0, 1)$. We set $V(\epsilon) = 1$ so that the regression coefficients approximate Cohen's d effect sizes. We consider the case with cross-over interaction where the optimal condition for each contextual group is different from the other one. In this settings, two distinct scenarios are covered; 1) **Context-only**: No main effect averaged across groups for text versus video hints, but the main effect within each contextual group is prominent (outcome generating model: $Y_i = 0 + 0.3D_i - 0.6D_iX_i + \epsilon_i$). 2) **Main-plus-context**: There is an average main effect in addition to the main effect for each group (outcome generating model: $Y_i = 0 + 0.3D_i - 1.2D_iX_i + \epsilon_i$).

For each scenario, we compare three policies: uniform random sampling, non-contextual and contextual multi-arm bandits. Non-contextual MAB does not contain a term that includes X , while contextual MAB includes this term (eq. 1). All results average 2500 repetitions of each simulation.

After the data are generated using one of the three assignment methods (e.g. uniform, non-contextual MAB, or contextual MAB), we fit an ordinary least squares (OLS) regression with terms for both effect of D and the interaction between D and X (the true outcome generating process shown in equation 1). Then for each treatment assignment policy and outcome generating scenario, we evaluate the i) Regret of action (difference in rewards between the optimal action and the assigned action), ii) Proportion of students assigned to the optimal condition for them, iii) Bias of coefficients of the fitted model (mean of estimated coefficients across simulations minus the corresponding true value), and iv) Proportion of simulations in which the OLS fitted model implies the true optimal policy.

4.2 Simulation Results

Context-only: No Main Effect. When there is no main effect, only the contextual MAB improves student outcomes. After only 250 students, the contextual MAB assigns an average of 77% of students to their optimal condition (Table 1). Because the non-contextual MAB cannot differentiate between individual students, both the non-contextual MAB and uniform random policy assign about 50% of students to their optimal condition.

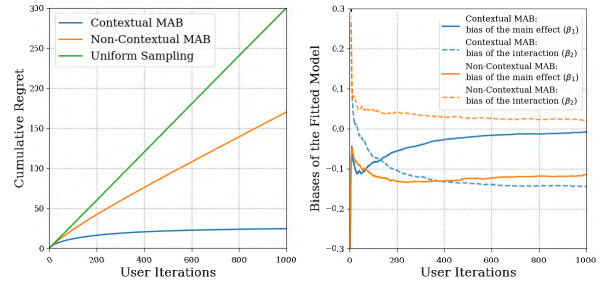


Figure 1: The cumulative regret at each iteration (left). The biases of the coefficients of the OLS model fitted on the generated data up to the specified user iteration (right). Outcome are generated by the model $Y_i = 0 + 0.3D_i - 1.2D_iX_i + \epsilon_i$ and averaged across 2500 repetitions.

The contextual MAB's adaptive assignments leads to systematic differences between the estimated model and the true data-generating model: the relative effectiveness of receiving text hints ($D = 1$) for students with high prior knowledge ($X = 0$) compared to lower prior knowledge ($X = 1$) is incorrectly estimated. After 1000 students, the estimate of β_2 is roughly 19% lower than its true value. Random variation in outcomes combined with the algorithm's adaptivity cause this overestimation of the effect size ($\beta_2 < 0$ and $\text{Bias}(\hat{\beta}_2) < 0$): the algorithm is more likely to stop assigning students with low prior knowledge to text hints when its benefit is below, rather than above, its expected value. Although bias is present, the resulting fitted model implies the correct optimal policy in 99% of repetitions.

Both the non-contextual MAB and the uniform random policy assign half of students to each condition, independent of prior knowledge, so both recover estimates of the effects that are quite close to the truth. However, the non-contextual MAB has more variance than the uniform random policy in its condition assignments, resulting in noisier estimated effects as reflected in the standard errors.

Main-plus-context: Main effect and larger effect for one subgroup. When there is both a main effect overall and a differential effect based on a student's prior knowledge, all three assignment methods differ in the degree to which they direct students to a more effective condition and in the biases in their estimates of the effects. As shown by lowered regret in Figure 1 (left), both MAB algorithms achieve better average student outcomes over the uniform random policy. Contextual MAB assigns students to conditions that are beneficial for them based on the feature value. However, non-contextual MAB cannot differentiate its policy based on students' prior knowledge. Since the overall main effect is negative (video hints are better than text hints on average), the non-contextual MAB learns to assign more students to videos over time. In smaller samples (left of Table 1), 82% of students with low prior knowledge are assigned to their best condition whereas only 18% of students with high prior knowledge are assigned to text hints although it is most beneficial for them. Thus, if heterogeneous effects are present, experimentation using a non-contextual MAB policy can be systematically worse for some students than the uniform random policy.

As in the **context-only** scenario, the contextual MAB overestimates the value of text hints for students with lower prior

Table 1: Assignment to Treatment Policies and Quantities of Interest

Simulations	Metrics	Participants 1 to 250			Participants 750 to 1000		
		Uniform	Non-Contextual MAB	Contextual MAB	Uniform	Non-Contextual MAB	Contextual MAB
Crossover Effect Only $Y_i = 0.3D_i - 0.6D_iX_i + \epsilon_i$ $\beta_1 = 0.3$ $\beta_2 = -0.6$	Regret	0.1497	0.1501	0.069	0.1503	0.1498	0.0071
	Prop. optimal action	0.5	0.5	0.77	0.5	0.5	0.98
	Prop. optimal action (X = 0)	0.5	0.49	0.76	0.5	0.49	0.99
	Prop. optimal action (X = 1)	0.5	0.51	0.78	0.5	0.51	0.96
	Bias($\hat{\beta}_1$) = $E(\hat{\beta}_1) - \beta_1$	-0.0019 (0.0032)	-0.0095 (0.0061)	-0.0193 (0.0046)	-0.0004 (0.0016)	-0.009 (0.0043)	-0.0003 (0.0021)
	Bias($\hat{\beta}_2$) = $E(\hat{\beta}_2) - \beta_2$	-0.0014 (0.0036)	-0.0019 (0.005)	-0.1093 (0.0073)	-0.001 (0.0018)	0.0011 (0.003)	-0.1142 (0.0052)
	Prop. of repetitions optimal policy is inferred	0.9388	0.7964	0.8904	1.0	0.918	0.9924
	Regret	0.2995	0.203	0.0703	0.2991	0.155	0.0039
	Prop. optimal action	0.5	0.5	0.83	0.5	0.5	0.99
	Prop. optimal action (X = 0)	0.5	0.18	0.73	0.5	0.02	0.99
Main and Crossover Effect $Y_i = 0.3D_i - 1.2D_iX_i + \epsilon_i$ $\beta_1 = 0.3$ $\beta_2 = -1.2$	Prop. optimal action (X = 1)	0.5	0.82	0.93	0.5	0.98	1.0
	Bias($\hat{\beta}_1$) = $E(\hat{\beta}_1) - \beta_1$	-0.0017 (0.0031)	-0.1336 (0.0082)	-0.0467 (0.0051)	0.0013 (0.0016)	-0.1153 (0.0065)	-0.0089 (0.0022)
	Bias($\hat{\beta}_2$) = $E(\hat{\beta}_2) - \beta_2$	0.0021 (0.0036)	0.0376 (0.009)	-0.1163 (0.0096)	0.0011 (0.0018)	0.0195 (0.0072)	-0.1445 (0.008)
	Prop. of repetitions optimal policy is inferred	0.9704	0.7516	0.9168	1.0	0.8	0.992

Notes: Standard errors for the biases of coefficients are shown in parenthesis.

knowledge ($\beta_1 + \beta_2 < 0$ and $\text{Bias}(\hat{\beta}_1 + \hat{\beta}_2) < 0$ in Table 1). The non-contextual MAB instead underestimates the effectiveness of text hints relative to video hints for students with high prior knowledge ($\beta_1 > 0$ and $\text{Bias}(\hat{\beta}_1) < 0$). Both patterns of bias occur because of the combination of random variation in outcomes combined with adaptive sampling, and the difference stems from whether the adaptive sampling is sensitive to contextual features. Prior work shows that less effective conditions tend to be underestimated by MABs, which usually leads to *overestimation* of effect sizes [5, 9]. Our results demonstrate that when heterogeneous effects are present but a non-contextual MAB policy is used, underestimation of the value of a condition that is worse on average can lead to *either* underestimation or overestimation of effect sizes for subgroups of students. When a condition is worse on average but better for some students, such as our text hints for students with high prior knowledge, the size of the effect for those students will be underestimated ($\beta_1 > 0$ and $\text{Bias}(\hat{\beta}_1) < 0$). The size of the effect of condition for the other students, such as our low prior knowledge students, will then tend to be overestimated ($\beta_1 + \beta_2 < 0$ and $\text{Bias}(\hat{\beta}_1 + \hat{\beta}_2) < 0$).

5. CONCLUSION

In this paper we focus on a scenario in which the impact of a binary condition assignment (text versus video hints) on a continuous student outcome depends on a binary contextual feature (low versus high prior knowledge). We find that assigning students to conditions using a contextual MAB significantly reduces regret relative to traditional RCTs and non-contextual MAB. Although adaptive, personalized experimentation can help more students benefit from experimental interventions and does accurately recover the direction of effects, the estimated differences of personalized effects across student subgroups will be biased. However for contextual MAB the presence of bias does not significantly impact the probability of inferring the optimal policy relative to a traditional RCT. Future directions include evaluating the effectiveness of methods to correct for the biases documented here, and using contextual MABs in real educational technologies to conduct adaptive experiments in settings where personalized treatment effects are expected.

6. ACKNOWLEDGEMENT

The authors gratefully acknowledge a grant from the **Office of Naval Research** (#N00014-18-1-2755).

7. REFERENCES

- [1] S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the 30th ICML*, volume 28, pages 127–135, 2013.
- [2] D. W. Braithwaite and R. L. Goldstone. Effects of variation and prior knowledge on abstract concept learning. *Cognition and Instruction*, 33(3):226–256, 2015.
- [3] B. Clement, D. Roy, P.-Y. Oudeyer, and M. Lopes. Multi-armed bandits for intelligent tutoring systems. *Journal of Educational Data Mining*, 7:20–48, 2015.
- [4] M. Dimakopoulou, S. Athey, and G. Imbens. Estimation considerations in contextual bandits. *arXiv preprint arXiv:1711.07077*, 2017.
- [5] A. Erraqabi, A. Lazaric, M. Valko, E. Brunskill, and Y.-E. Liu. Trading off rewards and errors in multi-armed bandits. In *Proceedings of the 20th Int. Conf. on AISTATS*, volume 54, pages 709–717, 2017.
- [6] D. N. Hill, H. Nassif, Y. Liu, A. Iyer, and S. Vishwanathan. An efficient bandit algorithm for realtime multivariate optimization. In *Proceedings of the 23rd SIGKDD*, pages 1813–1821. ACM, 2017.
- [7] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pages 661–670. ACM, 2010.
- [8] Y.-E. Liu, T. Mandel, E. Brunskill, and Z. Popovic. Trading off scientific knowledge and user learning with multi-armed bandits. In J. Stamper, Z. Pardos, M. Mavrikis, and B. McLaren, editors, *Proceedings of the 7th International Conference on EDM*, 2014.
- [9] A. N. Rafferty, H. Ying, and J. J. Williams. Bandit assignment for educational experiments: Benefits to students versus statistical power. In *Proceedings of the 19th International Conference on AIED*, 2018.
- [10] N. Sawant, C. B. Namballa, N. Sadagopan, and H. Nassif. Contextual multi-armed bandits for causal marketing. *ICML’18 Workshops*, 2018.
- [11] J. J. Williams, A. N. Rafferty, D. Tingley, A. Ang, W. S. Lasecki, and J. Kim. Enhancing online problems through instructor-centered tools for randomized experiments. In *Proc. of CHI*. ACM, 2018.

Investigating effects of considering mobile and desktop learning data on predictive power of learning management system (LMS) features on student success

Varshita Sher
School of Interactive Arts and
Technology
Simon Fraser University,
Canada
vsher@sfu.ca

Marek Hatala
School of Interactive Arts and
Technology
Simon Fraser University,
Canada
mhatala@sfu.ca

Dragan Gašević
Faculty of Education
Monash University, Australia
dgasevic@acm.org

ABSTRACT

The research area of analyzing log file trace data to build academic performance prediction models has tremendous potential for pedagogical support. Currently, these learner models are developed from logs that are composed of one intermixed stream of data, treated in the same manner regardless of which platform (mobile, desktops) the data came from. In this paper, we designed a correlational study using log data from two offerings of a blended course to investigate the effects of the variables, derived from the use of varying platforms, on the prediction of students' academic success. Given that learners use a combination of devices when engaging in learning activities, it is apparent that weighing the logs based on the platform they originate from might generate different (possibly better) models, with varying priority assigned to different model features. For instance, our results show that the overall frequency of course material access is a less powerful indicator of academic performance compared to the frequency of course material access 'from mobile devices', probably due to the benefits associated with ubiquitous any-time access available to mobile learners. Thus, the primary goal of this study is to bring to light the potential for improvement of prediction power of models after considering the learner's platform of access, within the learning analytics community and the fields of user modeling and recommender systems, in general.

Keywords

Learner Models, Learning Success, Learning Analytics, Mobile Learning

1. INTRODUCTION

The performance prediction models use students' logs from various learning activities that are available for measurement such as logging in, reading files, viewing posts, posting discussions and accessing feedback. However, research has

provided evidence suggesting not all activities (features) are equally effective as predictors of outcomes [3]. Moreover, research has also suggested that not all the learning activities are performed using a single technological modality [4, 6] but are often interleaved between devices such as mobile and desktop. In other words, depending on the utility and preference for a modality, the predictive power of learning indices (variables describing the frequency and/or quality of interaction with the LMS tool) in a regression model could be positively or negatively impacted. Building upon these inferences, we further posit that acknowledging the differences in the source of the log trace data used for modeling and predicting academic success, would promote increased accuracy of prediction models and explain anomalies. This hypothesis is supported by the results from a recent study [5] where the authors found a significant impact of the students' adopted platforms (and patterns of usage) for various learning activities on the final course grade.

The review of the literature reveals that the performance prediction models draw benefits from the students' 'event-driven logs' [1] from various learning activities that are available for measurement in a web-based learning management systems (LMSs) such as logging in, reading files, viewing posts, posting discussions and accessing feedback; all of which provide early indicators of student academic performance [8, 9, 2]. These logs, however, are composed of one intermixed stream of data, treated in the same manner regardless of which modality (mobile, desktops) the data came from. As a rule of thumb, the data concerning each predictor action, such as posting discussions and viewing course videos – actions that more often than not, emanate from different modalities and last for different durations – is generally pooled across all modalities. For instance, the frequency of access to course material from desktops, mobiles and tablets is typically used in the predictive model as one cumulative count measure i.e. *course_material_access*, counting all occurrences of course material access in the log file. This is done mainly due to the lack of awareness regarding the utility of technological context or merely to facilitate ease of data processing. Either way, the omission of technological modality variables in a model has potential to, at minimum, discard some useful information and as a result lower the prediction accuracy of the model, or more critically, cause serious threat to its interpretation. Thus, the primary aim of this paper is to create awareness of the role of modalities

Varshita Sher, Marek Hatala and Dragan Gasevic "Investigating effects of considering mobile and desktop learning data on predictive power of learning management system (LMS) features on student success" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 651 - 654

in predictive analyses of academic performance.

The exploration of the impact of modalities on predictive analytics is justified and highly recommended since: (a) learning activities are often completed by students using multiple modalities, used either sequentially or simultaneously [4, 7], and (b) identification of modalities that are ill-aligned to a task is important as they could undermine knowledge construction and may lead to unintended consequences in academic outcomes [5]. This paper thus investigates the usefulness of a modality-inclusive learner model, over and above a generalized model, for predicting learner success (operationalized by academic performance).

2. METHODOLOGY

2.1 Study Design

The study follows a correlational design as it investigates the effects of the variables derived from the trace data from different modalities, on the prediction of student's academic success, operationalized via percent mark - a continuous variable ranging from 0% to 100%. The data was collected over two semesters (Fall 2017 and Fall 2018) from two subsequent offerings of the same course. The course lasted 13 weeks and had a combined enrolment of 165 students (83+82). The course used blended delivery, utilizing the university's learning management system (LMS) to support learning activities and students' overall schoolwork. In addition to the web-browser versions of the LMS (desktop/laptop/ mobile), students had access to the mobile app version provided by the LMS vendor. Upon comparison of the features and functionalities offered by the two versions, no apparent differences were revealed. In the next section, we describe the various kinds of variables that were derived from log files.

2.2 Feature Engineering from LMS trace data

To investigate the effect of modality on different types of commonly included learning-related activities and their traces in the online courses, we selected 10 features (5 counts + 5 time spents for each activity) for inclusion in our analyses as predictors of academic success. Variables derived from the LMS trace data include information about the usage of the following tools/features: syllabus, course material (lecture + tutorial slides and instructor provided supplementary material), assignments, feedback on the assignments and calendar. Table 1 contains the types and total counts of learning actions, categorized into activities, captured by the LMS.

Table 1: Breakdown of activities and access (in terms of the number of actions) from different modalities.

Activity	Desktop	Mobile	Tablet
Assignments	15,929	2,474	23
Calendar	1,734	4,687	43
Course Material	24,850	1,279	147
Submission Feedback	1,954	2,968	6
Syllabus	1,952	155	8

Next, for each student we extracted the number of times and the time spent on using a particular feature by aggregating individual operations such as adding student's assignment views across all four assignment tasks to compute *count_assignment*. We call these variables *LMS features*. Each of these variables was split up further to account for the platform used to access that particular feature. For instance, in addition to having total number of assignment views for a student, we compute three more variables - mobile views, desktop views and tablet views- which indicate the respective number of course logins from each of the three main modalities. We call such variables *Modality features*.

The trace data for both LMS and modality features were initially collected as continuous variables. However, tablets were not used by many students, and therefore variables accessed from tablets were dichotomized into the *Accessed* and *Did not access* categories. Additionally, highly skewed variables were transformed using Box-cox transformations to correct their skewness. If the skewness still persisted, they were transformed into categorical variables and the cut-offs were decided arbitrarily to best represent the data.

2.3 Statistical Analyses

For each of the ten learning features introduced in Section 2.2, two regression models (Figure 1) were built using (a) LMS action variable, i.e. LMS feature (Model 1: simple linear regression), and (b) LMS action variable with information on modality source, i.e. Modality features (Model 2: multiple linear regression), to assess the importance of the platform source of the log data for predicting student percent marks. For each of the ten features, a change in R^2 from Model 1 to Model 2 is calculated to present the percentage of variability in student percent mark explained by Modality features over and above the LMS features. To ascertain whether the change was statistically significant, an ANOVA analysis using F-test of the statistical significance of the increase in R^2 was conducted.

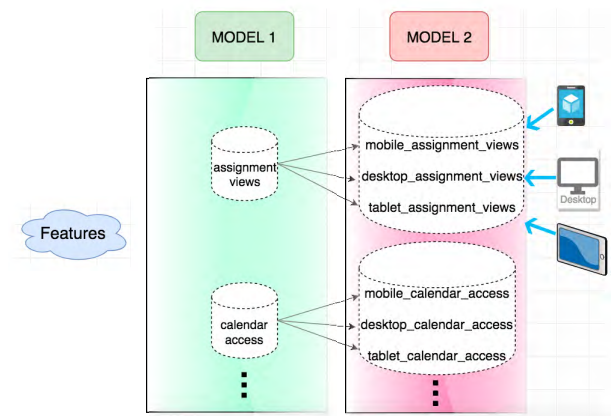


Figure 1: Feature selection for the two models.

3. RESULTS AND DISCUSSION

The results of the regression models featuring the associations between students' use of features from logged data - calculated cumulatively vs. partitioned based on the modality - and student course grades are presented in Table 2, along with the subsequent model comparisons using ANOVA analyses (columns 5-6 in Table 2).

Table 2: The association between the variables of students’ use of the LMS and Modality features and ln (log natural) student course grades: results of multiple linear regression models.

Activity (a)	Measure (m)	Model 1 $R^2 \times 100$ (p value)	Model 2 $R^2 \times 100$ (p value)	F-value	p-value	Modality features	β Coefficients
Syllabus	count	1.4% (p = 0.12)	3.9% (p = 0.03)	3.16	0.041	Desktop_Accessed 6-15 times vs. up to 5 times	12.36
						Desktop_Accessed more than 15 times vs. up to 5 times	31.86
						Mobile_Accessed 1-2 times vs. Did not Access	-23.11
	time spent	0% (p = 0.56)	1.4% (p = 0.18)	2.59	0.078	Mobile_Accessed more than 2 times vs. Did not Access	-8.35
						Desktop_Accessed (20, 40] hours vs. up to 20 hours	38.90
						Desktop_Accessed more than 40 hours vs. up to 20 hours	4.90
Assignment	count	7.3% (p < 0.001)	11.5% (p < 0.001)	2.92	0.023	Desktop_Accessed 51-80 times vs. up to 50 times	40.43
						Desktop_Accessed 81-100 times vs. up to 50 times	66.56
						Desktop_Accessed more than 100 times vs. up to 50 times	73.53
	time spent	9.5% (p < 0.001)	13.4% (p < 0.001)	8.20	0.004	Mobile_Accessed 1-10 times vs. Did not Access	-10.91
						Mobile_Accessed 11-20 times vs. Did not Access	6.54
						Mobile_Accessed more than 20 times vs. Did not Access	7.24
Submission Feedback	count	2.9% (p = 0.03)	8.8% (p < 0.001)	6.18	0.002	Desktop_Accessed 51-80 times vs. up to 50 times	40.43
						Desktop_Accessed 81-100 times vs. up to 50 times	66.56
						Desktop_Accessed more than 100 times vs. up to 50 times	73.53
	time spent	3.2% (p = 0.01)	5.1% (p = 0.005)	4.21	0.041	Mobile_Accessed 1-10 times vs. Did not Access	-10.91
						Mobile_Accessed 11-20 times vs. Did not Access	6.54
						Mobile_Accessed more than 20 times vs. Did not Access	7.24
Calendar	count	1.4% (p = 0.50)	1.9% (p = 0.92)	0.16	0.976	In assignment time_Desktop	0.71
						Mobile_Accessed up to 1 hour vs. Did not Access	17.39
						Mobile_Accessed (1, 2] hour vs. Did not Access	60.31
	time spent	0.5% (p = 0.80)	2.1% (p = 0.74)	0.85	0.468	Mobile_Accessed more than 2 hours vs. Did not Access	5.13
						Desktop_Accessed 11-20 times vs. up to 10 times	26.55
						Desktop_Accessed more than 20 times vs. up to 10 times	55.08
Course Material	count	0.3% (p = 0.20)	1.6% (p = 0.18)	1.52	0.198	Mobile_Accessed 1-10 times vs. Did not Access	3.28
						Mobile_Accessed more than 10 times vs. Did not Access	0.66
						In submissionfbk time_Desktop	4.68
	time spent	1.9% (p = 0.04)	0.7% (p = 0.24)	0.01	0.989	In submissionfbk_time_Mobile	0.98
						Desktop_Accessed 1-10 times vs. Did not Access	-8.11
						Desktop_Accessed more than 10 times vs. Did not Access	1.63

Based on our results of the multiple regression models, we can confirm that the choice of modality for a particular activity in a learning environment plays an important role in the overall model fit and subsequent modal interpretation. The significant ANOVA results imply that an increased proportion of variability in student course grades can be explained if the activity measures are calculated across modalities (Model 2) instead of using one cumulative measure (Model 1).

Interestingly, there was a notable difference in the impact (positive or negative) on the students’ course grades explained by the type of modality – desktop vs. mobile – used to perform the activity. For example, the results of the multiple linear regression analyses performed on the time spent on syllabus access indicated that the mobile access was a significant predictor of student learning outcome whereby course grades of students who used mobile phones for substantive duration (1-2 hours) to access the syllabus were about 13% *lower* than those of their counterparts who did not spend any time accessing the syllabus from the mobile phone modality ($\beta = -12.9$, $p = 0.04$). On the contrary, looking at the time spent on viewing the course assignments, the mobile phone modality reflected a positive association with course grades and explained a greater amount of vari-

ance, such that the course grades of students who used mobile phones to view the assignments for 1-2 hours were about 60% *higher* than those of their counterparts who did not use the mobile phone modality at all ($\beta = 60.3$, $p = 0.01$).

More importantly, the impact of these modalities in explaining the overall fit was not consistent across activities in the learning environment, both in their presence and magnitude. That is to say, *some modalities may or may not play a role in determining student’s course grade depending upon the activity performed using the modality*. For instance, the duration of time spent on a desktop for viewing the assignments was a significant predictor of student course grades whereby a 10% increase in time spent resulted in around 7% increase in student course grades. On the contrary, the same modality was not significant at all when the activity involved engaging with the course material. However, the desktop modality was again found significant for the submission feedback activity where this effect was seven times larger compared to assignment viewing i.e. a 10% increase in time spent on engaging with the feedbacks on assignment submissions resulted in around 47% increase in student course grades.

4. LIMITATIONS FUTURE WORK

Since the feature space in our study is high-dimensional and could easily include interactions and non-linear effects, our immediate next steps involve comparison of machine learning classification methods to test the same hypotheses thoroughly. To further broaden the discussion, there are in fact many features that can be computed from trace data and that are used in the prediction models. As we saw improvements in the ‘crude’ features that we investigated, it is conceivable that we can see improvement in other derived features and therefore improve fidelity of the models.

Furthermore, it would also be interesting to devise, using a bigger participant pool and diverse activity pool, the most optimal learner model comprising a combination of highly explanatory LMS and Modality features from various learning activities as predictors. This, in turn, would require knowledge of several activities in a learning environment for which, modality features can explain more variance in learning outcomes compared to standard cumulative LMS features.

Our methodology involved tracking user interaction with the LMS and this may raise a concern about the extent to which our results were dependent upon the activities targeted in the LMS and the design of the LMS (both browser and app) itself. The types of activities included in our study are quite common in instructional design and usually captured in the same way, thereby rendering good generic results. However, there might be variances in how learning activities are structured and presented in LMS and some LMS can offer even more fine grained tracking to see the influence of modality features from various other activities on the learning outcomes.

5. CONCLUSIONS

Taking up the research on use of mobile and desktop devices in learning environments and its ramifications on learning outcomes one step further, in this paper we looked at how modalities used by students for carrying out learning-related activities in the LMS, could act as powerful indicators of academic success. We designed separate prediction models using measures (e.g., counts and time spent) of activities in the learning environment aggregated across (a) all log data and (b) each individual modality in the log data. We observed that acknowledging a learner’s modality context – i.e. a dynamic entity constructed by the learner through interaction with the learning management system from desktop and mobile devices – led to improvements in the accuracy of models.

To further illustrate the significance of these improvements in predictive power, statistical analyses confirmed the improvements to be significant for most of the predictor measures assessed in this study. While the magnitude of improvements may not be of particular interest, the major take away from the study is that interpretations and subsequent interventions based off of generalized learner models may be improved by utilizing modality-inclusive models, since modalities contribute differently to the learning process depending on the activity they are used for. Further, the significance of this research lies in the simplicity of the method by which the modality of access for a learning ac-

tion/activity can be readily available through capturing the ‘user-agent’ from the students’ log data and the potential high impact it has on the prediction process.

The major highlights from the paper include:

1. Tracing the modality source of log data improves accuracy of learner models
2. Some modalities are better predictors of learning outcomes than others
3. Magnitude of outcome variance explained by modality differs based on the activity
4. Direction of outcome variance explained by modality differs based on the activity

6. ACKNOWLEDGMENTS

This research was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and Social Sciences and Humanities Research Council of Canada (SSHRC).

7. REFERENCES

- [1] C. Brooks and C. Thompson. Predictive Modelling in Teaching and Learning. In C. Lang, G. Siemens, A. F. Wise, and D. Gašević, editors, *The Handbook of Learning Analytics*, pages 61–68. Society for Learning Analytics Research (SoLAR), Alberta, Canada, 1 edition, 2017.
- [2] J. P. Campbell, P. B. DeBlois, and D. G. Oblinger. Academic analytics: A new tool for a new era. *EDUCAUSE review*, 42(4):40, 2007.
- [3] D. Gašević, S. Dawson, T. Rogers, and D. Gasevic. Learning analytics should not promote one size fits all: The effects of instructional conditions in predicting academic success. *The Internet and Higher Education*, 28:68–84, 2016.
- [4] G. E. Krull. *Supporting seamless learning: Students’ use of multiple devices in open and distance learning universities*. PhD thesis, 2017.
- [5] V. Sher, M. Hatala, and D. Gašević. On multi-device use: Using technological modality profiles to explain differences in students’ learning. In *Proceedings of the Ninth International Conference on Learning Analytics & Knowledge*. ACM, 2019, in press.
- [6] G. Stockwell. Using mobile phones for vocabulary activities: Examining the effect of platform. 2010.
- [7] G. Stockwell. Tracking learner usage of mobile phones for language learning outside of the classroom. *CALICO Journal*, 30(1):118–136, 2013.
- [8] A. Y. Wang and M. H. Newlin. Predictors of performance in the virtual classroom: Identifying and helping at-risk cyber-students. *THE Journal (Technological Horizons In Education)*, 29(10):21, 2002.
- [9] A. Y. Wang, M. H. Newlin, and T. L. Tucker. A discourse analysis of online classroom chats: Predictors of cyber-student performance. *Teaching of Psychology*, 28(3):222–226, 2001.

Investigating Error Resolution Processes in C Programming Exercise Courses

Yuta Taniguchi
Faculty of Information Science
and Electrical Engineering
Kyushu University
Fukuoka, Japan
taniguchi@ait.kyushu-
u.ac.jp

Atsushi Shimada
Faculty of Information Science
and Electrical Engineering
Kyushu University
Fukuoka, Japan
atsushi@ait.kyushu-
u.ac.jp

Shin'ichi Konomi
Faculty of Arts and Science
Kyushu University
Fukuoka, Japan
konomi@artsci.kyushu-
u.ac.jp

ABSTRACT

This study investigates how we can understand students' actual status in C programming exercises from their learning activity logs. In a face-to-face course of C programming exercise, it is hard for a teacher to see who are in trouble from their appearance. It is not always true that typing something means he or she is making some progress. Therefore it is important to identify, or possibly even predict, students having difficulty from their activity patterns. Most of the prior work paid attention to only trial-and-error activities, such as compile results and execution errors. However, it tends to be overlooked that knowledge acquisition process is also worthy of attention. When a student encounters a compile error, they usually read textbooks to seek a solution. It is considered to be useful for the task whether he or she has an ability to find appropriate pages for error resolution. In this paper, we propose a method to predict whether a student can resolve errors or not. Based on students' activity logs collected from our programming environment and e-book system, we conduct experiments to show and discuss the prediction performance.

1. INTRODUCTION

The C programming language has many obstacles, such as relatively complex syntax and confusing messages of compile errors. In C programming courses, students are required to overcome those hurdles as well as to learn how to solve problems computationally. Especially, because error messages are not necessarily straightforward, students are sometimes led to irrelevant pages of textbooks and get confused. Such learning experiences perhaps lower their motivation to learn, and therefore teachers need to intervene to help students in such situations. However, not all students ask their teachers or friends when they have a tough issue, and it is hard for teachers to distinguish students in trouble from ones without problems by physical appearance. Hence data-driven

approaches have been considered for identifying such *at-risk* students.

Blikstein [1] and Helminen et al. [4] performed a kind of offline analysis of students' behaviors. On the other hand, a realtime-oriented analysis was performed by Fu et al. [3]. They proposed a dashboard system for teachers to grasp the current status of all students in real-time fashion. However, none of them considered the knowledge acquisition process. Helminen et al. [4] addressed the process where students struggle to resolve errors. However, in their study, only limited activities, e.g. selecting, ordering, and indenting code fragments, are analyzed, and activities such as referring external learning materials are not considered.

For understanding students' learning processes, it is significant to know how students search learning resources for necessary information and acquire knowledges. We believe students have to learn how to resolve errors by themselves, and intervention by teachers should be controlled. Hence we have to care how much effort was made by a student to obtain necessary knowledges for resolving errors. Although we could quantize such efforts to some extent by observing how they use their textbooks during programming exercises, only a limited number of studies focused on students' trial-and-error and knowledge acquisition in learning processes of programming languages.

Toward automatic and real-time detection of students in need of help, in this paper, we analyze *error resolution processes* in which students struggle to resolve compilation errors with course materials in a programming exercise course. Furthermore, we try to predict whether they can successfully resolve errors or not in early stage of error resolution processes. Toward this end, we employ both compilation logs and page view logs of e-textbooks, and characterize compilation errors in relation to exercise questions and individual students.

2. METHODS

2.1 Error Resolution Processes

The C Programming Language belongs to the compiled languages, which need a compile process ahead of executing a program. In the compile process, a compiler program transform input source code into a machine code. The process involves a lot of checks and many problems are found as *er-*

Yuta Taniguchi, Atsushi Shimada and Shin'ichi Konomi "Investigating Error Resolution Processes in C Programming Exercise Courses" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 655 - 658

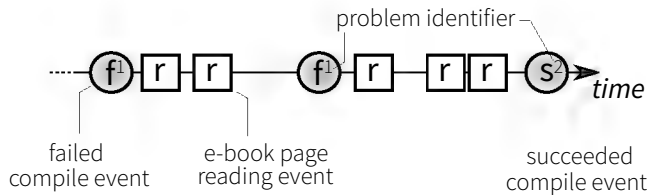


Figure 1: An example of a timeline. A timeline may consist of three types of events: failure compile event, success compile event, and e-book page reading event.

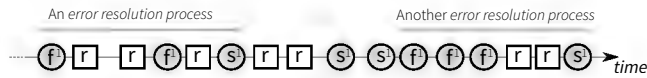


Figure 2: An example of an error resolution process found in a timeline. Every ERP starts with a failure compile event, but does not necessarily end with a success compile event.

rors before running a program. Errors found in this step could be typically divided into syntax errors and linker errors. The former type of errors requires a student to modify their source code so that it follows the syntax of C language. In the latter case, it usually occurs because of missing libraries and one need to specify required libraries on the command line (or a compiler's configurations). In this study, we call such a process as an *error resolution process (ERP)*, and analyze how students resolve such errors by themselves during exercises.

A sequence of a student's activities is called a *timeline*. A timeline consists of events, and we consider three kinds of events in this study: *failure compile event*, *success compile event*, and *e-book reading event*. The first two types occur when a student compiles his or her source code. If a compile finishes without printing any messages on a screen, we consider it is a success compile event. On the other hand, it is considered as a failure compile event if a compiler outputs some messages. An e-book reading event indicates a student started to read a page of an e-book.

These concepts are illustrated in Figure 1. The figure shows a sequence of eight events from left to right. Every compile events are associated with a problem that a student try to solve. Identifiers of problems are shown at the top right of each event. For this example, one of the possible interpretations of the timeline is as follows; a student were working on the problem 1 and compiled their program for it; however, they got an error from the compiler, and then they rewrite a program according to textbooks and tried again; unfortunately, it did not work well, and he or she abandoned the problem and began the problem 2; they compiled another program, and it successfully finished.

We more precisely describe an ERP based on a timeline. An ERP is a contiguous subsequence of a timeline; an ERP always starts with a failure compile event, and it can include only compile events associated with the same exercise problem. For example, Figure 2 shows two ERPs within a timeline. Since the timeline does not involve another problem in

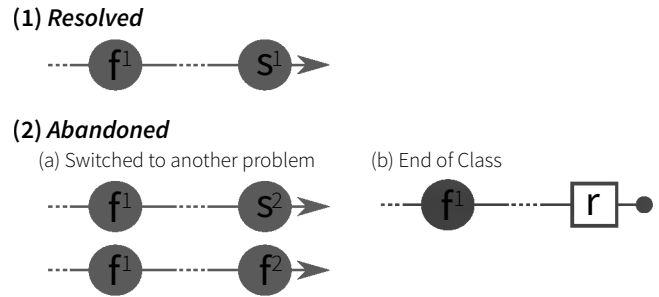


Figure 3: Possible cases of ERPs whose result could be either *resolved* or *abandoned*.

this case, we can simply identify ERPs as longest contiguous subsequences starting with a failure compile event and containing at most a single success compile event at the end of the subsequence. Another example is shown in Figure 1 which consists of all except the last event.

While several scenario could be considered for ERPs, we simply consider two kinds of outcome from an ERP: *resolved* and *abandoned*. Figure 3 shows examples for both cases. A resolved ERP ends with a success compile event associated with the same problem as the ERP's initiating failure compile event. An abandoned ERP could finish with either an e-book event or a failure compile event. The latter type of ERPs appears when a student switch a current exercise problem to another or a class is over.

2.2 Predicting Outcomes from ERPs

It might be an important information for teachers whether a student's ERP will end up with *resolved* or *abandoned* state. We consider the prediction of the outcome from an ERP in early stage of the process. To characterize EPRs, we employ n-gram features. Firstly, we obtain a textual representation of an ERP. In the representation, every event is notated as a single letter. In this paper, we use *s* for a success compile event, *f* for a failure compile event, *c* for an event of reading e-textbooks especially for the programming course, *o* for an reading event but for other course materials. Furthermore, we also encode gaps between events with - which denotes a gap of just 10 seconds. Any gaps less than 10 seconds are ignored. For example, a gap of 24 seconds is notated as --.

N-gram features are then extracted from the text representation of a timeline. As there are five letters in the representation, an n-gram feature vector will be a 5^n -dimensional vector. For example, if we use 3-gram-based features, a student is represented as a 125-dimensional vector. In this study, we combine 1- to 4-gram feature vectors into a single feature vector, i.e. $5^1 + 5^2 + 5^3 + 5^4 = 780$ -dimensional vector, and use it for making prediction.

Students' first actions are considered important; for example, an expert will read error messages carefully, fix problems, and then recompile the program while a beginner will just repeat the compile or read many pages of textbooks seeking for a solution. We expect such differences are captured by n-gram representations of the first t minutes.

In our experiments, we targeted only ERPs longer than

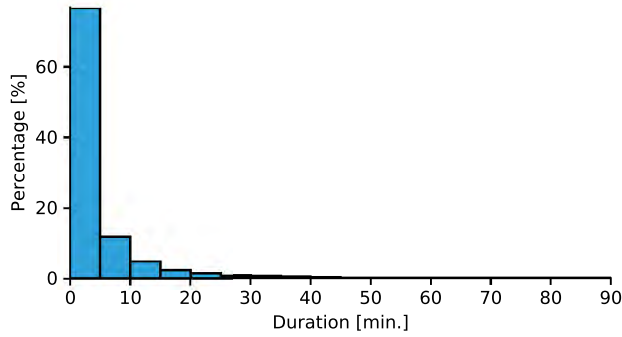


Figure 4: The distribution of durations of error resolution processes.

Table 1: Hyperparameters adjusted.

Hyperparameter	Candidates
criterion	gini, entropy
max depth	1, 2, 3, 4, 5, 6, 7, 8
max features	sqrt, log2, None

$2t$ minutes. If we use the first t minutes for predicting the outcomes of ERPs whose durations are less than $2t$ minutes, it means that we use more than a half of information to predict the final status of ERPs. That is why we exclude ERPs shorter than $2t$ minutes in this study.

We set $t = 5$ in our experiments. There are 29,216 ERPs in our dataset. Figure 4 shows the distribution of the durations ERPs took. The number of ERPs which only took less than or equal to 10 minutes was 25,886 (88.6%), and the number of our target ERPs was 3,330 (11.4%).

For the prediction, we employ random forest classifiers as a classifier. Random forests are one of the ensemble learning algorithm based on bagging technique and random selection of features [2]. This is one of the most popular classifiers not only in educational datamining community but also in wider data science community. It is known that the classifier is robust even when each dimension of feature vectors has different scales. Therefore, we simply apply random forests to n -gram feature vectors without normalizing them.

We use the implementation of random forest classifier included in scikit-learn package [6] in this study. Basically, we use the default values of the library for the classifier’s hyperparameters except for ones shown in Table 1 and the number of estimators which was set to 100. We optimize these parameters by a grid search algorithm, whose implementation is also provided by scikit-learn. The candidate values for these parameters are shown in the table.

2.3 Data Collection

The course we target is the introductory courses for C programming language in our university. Primarily, all freshmen students takes the course. The class is composed of lectures and coding exercises. There are about 20 classes for the course for each year, and almost all of the courses are taught by different teachers. Although it is not enforced, we

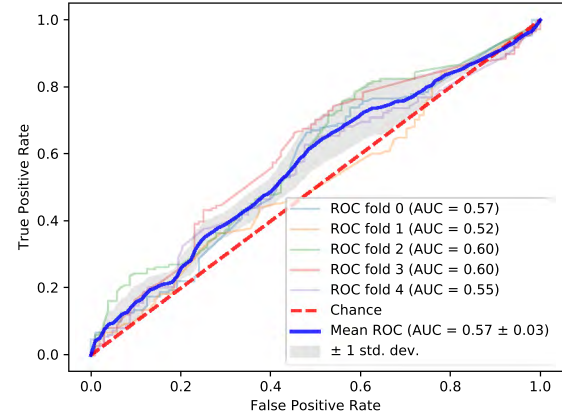


Figure 5: The ROC curve obtained from 5-fold cross validation.

have a set of standard course materials, and they are used in almost all classes.

In exercise, we use the compiler program “gcc”, from the GNU compiler collection. The compiler program is modified from the original version so that it can record students’ learning logs. More precisely, when it is executed, it saves given commandline arguments, the contents of given source files, and the output of the compiler as well as the time and student IDs. Since a commandline and source code are available as logs, we can reproduce what a student tried and what he or she obtained as a result. From those learning logs, we reconstruct compile events in timelines.

We also utilize students’ learning logs on our own web-based e-book system BookRoll [5]. All the course materials are available on the system, and therefore we can collect students’ learning logs about how student utilized those textbooks during exercises. Students’ actions, such as flipping pages and adding highlights, on the system are collected immediately as events occur. In this study, we only focus on page-flipping events collected for our analysis.

3. RESULTS & DISCUSSION

We evaluated our prediction performance using AUC metric, which stands for area under the ROC curve, combining with 5-fold cross validation. The hyperparameter values obtained by the grid search algorithm were `criterion=gini`, `max_depth=6`, `max_features=sqrt`, `n_estimators=10`. Figure 5 shows the five ROC curves obtained during cross validation and the mean ROC curve computed from them.

Table 2 shows the average performance scores with four evaluation metrics. According to the table, we cannot say the performance is good. Although the recall value is relatively high, precision value (especially in the test phase) is low. AUC value for the test phase is better than the chance rate, it is not good enough for practical use.

Table 2: Cross validation result. Values are averaged over five folds.

Metric	Value
AUC (test)	0.566
F1 (test)	0.698
Precision (test)	0.598
Recall (test)	0.844
AUC (train)	0.684
F1 (train)	0.740
Precision (train)	0.640
Recall (train)	0.882

4. CONCLUSION

Focusing on students' error resolution processes, we proposed a prediction method for outcomes from those process. We encode event sequences into texts, and characterize them using n-gram features. From our preliminary analysis, the proposed method could not show a good performance while it has a little bit better performance than the chance rate.

Future work include improvement of the feature representation of timelines and further analysis on the characteristics of student's activities and abilities.

5. ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number JP17K12804.

6. REFERENCES

- [1] P. Blikstein. Using learning analytics to assess students' behavior in open-ended programming tasks. In *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*, pages 110–116. ACM, 2011.
- [2] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.
- [3] X. Fu, A. Shimada, H. Ogata, Y. Taniguchi, and D. Suehiro. Real-time learning analytics for c programming language courses. In *Proceedings of the Seventh International Conference on Learning Analytics & Knowledge*, LAK '17, pages 280–288, New York, NY, USA, 2017. ACM.
- [4] J. Helminen, P. Ihantola, V. Karavirta, and L. Malmi. How do students solve parsons programming problems?: An analysis of interaction traces. In *Proceedings of the Ninth Annual International Conference on International Computing Education Research*, ICER '12, pages 119–126, New York, NY, USA, 2012. ACM.
- [5] H. Ogata, Y. Taniguchi, D. Suehiro, A. Shimada, M. Oi, F. Okubo, M. Yamada, and K. Kojima. M2b system: A digital learning platform for traditional classrooms in university. *Practitioner Track Proceedings of the Seventh International Conference on Learning Analytics & Knowledge*, pages 155–162, 2017.
- [6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Staying in the Zone: Sequencing Content in Classrooms Based on the Zone of Proximal Development

Oded Vainas*
SimilarWeb
odedvainas@yahoo.com

Yossi Ben-David
Microsoft
bendavidyossi@gmail.com

Ran Gilad-Bachrach
Microsoft
rang@microsoft.com

Meitar Ronen
Microsoft
t-merone@microsoft.com

Ori Bar-Ilan*
Zeitgold
oribarilan@gmail.com

Roi Shillo
Ben-Gurion University
shillo@post.bgu.ac.il

ABSTRACT

Vygotsky's notions of Zone of Proximal Development and Dynamic Assessment emphasize the importance of personalized learning. In this work, we introduce a novel adaptive learning engine named E-gostky that builds on these concepts. E-gostky creates a machine-learning-based skipping policy that presents students with questions which will challenge, but not overwhelm them, keeping students in their Zone of Proximal Development. We evaluated our engine in a real classroom environment, including hundreds of students from several elementary schools. Our results show that using E-gostky can significantly reduce the time required to reach comparable performance. Specifically, in our experiment, it took students who were using the adaptive learning engine 17% less time to reach a similar level of mastery as of those who didn't. Moreover, students made greater efforts to find the correct answer rather than guessing, and class teachers reported that students showed higher engagement.

Keywords

Personalized learning path, e-learning, Adaptive learning, Zone of Proximal Development

1. INTRODUCTION

In the early 1930s, Lev Vygotsky conceptualized the idea of *Zone of Proximal Development* (ZPD), and laid the foundations for the concept of personalized learning [14, 4]. ZPD is defined as "the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem-solving under adult guidance, or in collaboration with more capable peers" [14, pg. 86]. In other words, it refers to what a learner can do with assistance, but cannot

do independently. According to Vygotsky, concrete growth can only occur in the ZPD, and the learning is most effective when the support is matched to the needs of the learner. However, creating a personalized learning environment is challenging in traditional classrooms, where the teacher is outnumbered by the students. Using modern technology in classrooms supports the teacher in ways that can allow effective personalized learning [5].

Many e-learning systems contain a large set of exercises¹ such that the students moves in a linear order between exercises. However, content served in a non-adaptive manner will under-challenge some students, and over-challenge others. In view of Vygotsky's theory, it can be said that the problem stems from the fact that each student has a different ZPD at a given point in time. The method proposed here tries to mitigate this problem using a quantitative interpretation of the ZPD concept, translating it to a data driven algorithm which supports the delivery of personalized learning content. The method utilizes an adaptive learning engine, named E-gostky, that can move forwards or backwards through linearly structured content to find the next exercise which will be at the right level of difficulty, keeping the student within her ZPD.

To identify whether a task will keep a student within her ZPD, we use another core concept of Vygotsky's theory: Dynamic Assessment (DA). The goal of DA is to assess the potential for learning, rather than a static level of achievement, by prompting students to use their minds and assistance in problem solving. By dynamically evaluating students' progress, we adapt the served content according to the learning potential of the student at the current time and on the current topic.

To test our method, we used content that was developed by pedagogical experts at the The Center for Educational Technology (CET) in Tel-Aviv, Israel, to serve elementary school students while learning fractions. When comparing the students who learned the content in its original form to students who learned using E-gostky, we found that students in both groups achieved comparable performance, yet to reach this

*This work was done while the author was with Microsoft

Oded Vainas, Yossi Ben-David, Ran Gilad-Bachrach, Meitar Ronen, Ori Bar-Ilan, Roi Shillo, Galit Lukin and Daniel Sitton "Staying In The Zone: Sequencing Content in Classrooms Based on the Zone of Proximal Development" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 659 - 662

¹The terms "exercise" includes also other activities such as watching instructional videos.

level of mastery, students on the adaptive track required significantly less time ($\sim 17\%$ less time on average)- time which can be used for other educational activities. Furthermore, the qualitative feedback provided by teachers suggests that students using E-gostky were more engaged and eager to learn. One of the teachers reported that “using the skipping engine (E-gostky) ... motivated them to learn, think longer before answering, and use the aids to answer correctly”.

In this document we present the summary of the results. A more comprehensive description of the study and the results can be found at [13].

2. BACKGROUND AND RELATED WORK

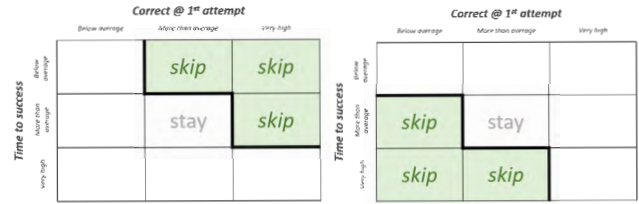
The Zone of Proximal Development (ZPD) plays a key role in the method we propose. The ZPD was introduced by Vygotsky with the goal of identifying the psychological functions (and related social interactions) that are needed for transitioning to the next age period, and to assess the current status of a child’s maturing functions [4]. The ZPD provides a framework to evaluate learners’ abilities, known as Dynamic Assessment (DA) [4, 8] which focuses on measuring the learning potential for future development [8]. A dynamic test is based on teacher assistance, where guidance, feedback and adaptive delivery of assistance are embedded in the evaluation procedure itself [4]. While originating from a developmental theory, many benefits of implementing the ZPD and DA concepts in educational applications have been demonstrated, such as increased motivation and higher performance [12, 15].

Our work also relates to past research on using historical data to sequence content for students, a problem that attracted many researchers [6, 11]. For a more comprehensive literature review see [13].

3. METHODS

The adaptive learning method we propose consists of two components: (i) *Dynamic Assessment* - this component predicts students’ performance in proposed exercises (ii) *Sequencing Policy* - this component is a policy in which given the predictions made by the Dynamic Assessment, assigns the next exercise by deciding which exercises are in the current ZPD of the student.

The Dynamic Assessment component is built on features learned by Random-Forests models [3] such as student history, offline exercise metadata and online student-exercises features. These features are then used to predict for each student-exercise pair, two factors: (1) *Time To Success (TTS)* - how much time will the student spend until she finds the correct solution to the exercise, and (2) *Correct at First Attempt (CFA)* - the probability that the student will solve the exercise correctly in her first trial [9, 10]. These predictions are then used to define a policy that is designed to keep students in their Zone of Proximal Development by ruling whether an exercise should be skipped. The skipping policy considers an exercise too easy for a student if she can solve it correctly and relatively fast (see Figure 1). More precisely, if the predicted CFA is above a certain threshold and the predicted TTS is below another threshold, then the policy rules the exercise as “below” the ZPD, thus too easy, and recommends skipping it. Similarly, the policy will skip



(a) Non-bonus exercises.

(b) Bonus exercises

Figure 1: The Zone of Proximal Development by CFA and TTS predictions. (a) demonstrate the stipulated ZPD and the skipping policy for regular (non-bonus questions), while (b) represents it for bonus questions.

a bonus exercise that is too challenging by the predictive models, to prevent the overwhelming of struggling students.

In addition, we adjusted E-gostky to handle “bad skips” (i.e. exercises that the policy would have skipped but the student had a hard time solving, meaning, not correct on the first attempt or took longer than the average time to solve the exercise). If a student does not solve correctly at first attempt an exercise that followed a skip, E-gostky sends her backwards in the curriculum, to the exercise that would have been presented, if there were no skips.

The policy also respects signals from content providers. For example, the content provider can flag an exercise as mandatory, in which case the exercise will be served to all students. Also, to improve future decisions and restrain the engine from making too many skips, with a certain probability, an exercise will be served to the student even if the policy described above suggests that it should be skipped.

To configure features the system uses at run-time, thresholds used for the skipping policy and the DA models were learned using offline data. This data was collected during the previous school year when 714 students were learning using the same content, but without the adaptive engine. We also used this data for the offline validation of the skipping policy, where we found that in comparison with other skipping policies (such as random skips and uniformly skipping after n consecutive successes), the chosen policy can save more time for the student while making fewer “bad skips”. During the offline evaluation, the thresholds of the system and the number of candidate exercises to skip were empirically selected. Tuning these parameters allowed adjusting the boundaries of the predicted ZPD. See [13] for more details about the methods used.

4. THE EDUCATIONAL CONTENT

The content was developed by pedagogical experts in the Center for Educational Technology (CET), with the goal of teaching fractions to 4th grade students. A learning path was provided, which is termed as the “baseline path”, designed by pedagogical experts. The content includes exercises, intertwined with interactive explanations and aids, and quizzes that are administered at the end of each section and used to assess mastery. A student can make as many attempts as needed to solve an exercise, and proceed to the next one.

5. IN-CLASS EXPERIMENT

The methods described above were implemented as a web-service that operates as follows. When a student solves an exercise, a request for the next exercise is sent from the content delivery system. E-gostky predicts the CFA and TTS for the following exercises and the policy is used to choose the next exercise to serve. The student is notified of the number of exercises skipped.

This service was deployed as part of an experiment including 412 students from 18 classrooms from 8 schools. Unfortunately, in 4 of the 8 schools, the data collected was invalid for the statistical analysis. This left 4 schools and 213 students for analysis purposes. The analysis considered the first two sections of the educational program where 60% of the students were using E-gostky and the rest were using the baseline approach as designed by the content provider.

6. EXPERIMENT RESULTS

In what follows, we compare the performance of students who learned using E-gostky, with those who learned with the baseline path. To control the classes' effect, we conducted a two-way ANOVA test. In cases where the assumptions to this test were not met, we used a Man-Whitney-U two-tailed test to compare both groups within each school separately and then used Bonferroni correction to get a corrected p -value for the entire test.

Total time spent. The total time spent is the sum of the time that the student spent on all of the exercises she was asked to solve. Students using E-gostky finished the exercises of the first two parts of the educational program 17% faster than the students in the baseline path (p -value = 0.025).

Time to solve an exercise. Students on the baseline path used less time on individual exercises but the difference is not statistically significant (p -value = 0.086).

Success Rate at First Attempt. In non-quiz questions, students' success rate in the adaptive path was significantly lower (p -value = 0.010) than that of the baseline students.

These results show that the E-gostky system worked as expected: skipping exercises that are likely to be answered correctly and fast to keep the students in their ZPD.

Quiz grades. To investigate E-gostky's effect on students' performance, we compared the students' average scores in two quizzes given to the students after completing the first and second units of the learning material. For quiz 1, the grades of the baseline group were higher, and this difference was borderline significant (p -value = 0.049) while in the second quiz there were no significant differences (p -value = 0.386). Since the groups were assigned randomly, it might be possible that the BL and ADL groups were not balanced. Hence, we evaluated the improvement rate between the quizzes. The adaptive group's mean improvement was higher, but only borderline significantly (p -value = 0.06).

A paired t-test was used to compare the probability to succeed on each question for each school separately. In both quizzes, the baseline path students' grades were significantly

higher (p -value = 0.008 \ 0.002 for the 1st \ 2nd quiz). However, when excluding School 4 from the analysis, the grade differences for quiz 1 are not significant (p = .108). This is not true for quiz 2 where in that case, the differences remain significant (p = .008).

The fact that the improvement rates are better for the treatment group (non significant) while the quiz scores are somewhat better for the control group may imply that the assignment of students to the treatment conditions was biased. Moreover, since some statistical tests show the differences in the scores to be significant while other tests do not, suggests that the differences, even if existent, are not big and may be controlled by tuning the skipping policy.

Guessing Rates. To compare guessing rates, we first set a time-limit threshold (see [13] for further details on the threshold selection). Questions which were answered faster than this threshold were regarded as guesses. Students using E-gostky guessed on average 25% less compared to the students in the baseline path. We found that both the treatment and the school had significant effects (p -value < 0.0001 in both cases), while the interaction effect was non-significant. The lower guessing rates in the adaptive path indicates that the students using E-gostky were more engaged, as suggested by previous studies [7].

6.1 Qualitative Results

Feedback from the teachers and students was gathered by face-to-face meetings or by phone interviews. The feedback reports reveal that the students enjoyed the overall experience of learning via a computerized environment. Besides that, they expressed special interest in working with the adaptive engine.

Engagement and motivation. The students reported that learning with E-gostky, which they nicknamed "the skipping engine", was more engaging and motivating, as two of the students mentioned, "I thought longer before answering each question, ... then I succeeded more and it skipped more questions for me...", "it gave me motivation to continue and I started to be more accurate...". This behavior was noticed by teachers: "There are children who came to class even in their free time".

Confidence and Self-efficacy Another important aspect of E-gostky is the immediate feedback the student received. It made them more confident about their proficiency and future success. "...It gave me a feeling that I know the material and that I can succeed each and every time", "... it gave me more confidence as I was advancing and the feeling that I have a better understanding of the material".

Usage of aids. Another effect of E-gostky was that to achieve more skips, the students showed a higher usage of aids to increase their accuracy in solving exercises. As one of the students reported, "...I used the lab more often and then I succeed more and then I got more skips".

7. DISCUSSION

The results show that students using E-gostky made similar progress between quizzes as did students in the control group. With that said, E-gostky saved ~ 17% of the time

for the students that were using it. The time saved, which is about $\frac{1}{6}$ of the time reserved for working with this e-learning system, can be used by teachers for other learning activities.

It might seem counter-intuitive at first glance that the students on the adaptive track spent more time on each exercise while their total time spent is smaller, but this is the result of E-gostky skipping over many easy exercises that the student would have solved fast. As Vygotsky's theory suggests, these exercises do not contribute to the learning process since they are not in the ZPD.

Another benefit of E-gostky is that the positive feedback students receive dissuades students from guessing, and motivates them to be accurate and as a result, further enhances learning [1, 2].

There are some limitations to the experiment we conducted. We see differences in the scores achieved in the first quiz between the groups. These differences are influenced by one particular school in which the differences are significant. However, we learned, by interviewing the teachers after the experiment, that some of them did not fully understand the adaptive mechanism and thought that exercises were skipped by mistake. This confusion was also reported in students' feedback: "At first I thought it was a bug in the software, so I went backwards to do skipped questions...", "In the beginning I thought there was something wrong with the system, so I went back to the questions it skipped...". We also note that some statistical tests indicate differences in the performance of the baseline group and the treatment group. However, there are indications that these differences may be due to biased assignments of students to the groups.

It is hard to distinguish between the contribution of exercise selection and the contribution of the notification students received about skipped exercises. Students learned that by providing accurate answers, the system will skip over exercises and this was considered as positive feedback. We conjecture that both factors contributed to the positive results of our system. However, quantifying the contribution is beyond the scope of this study.

8. CONCLUSIONS

We developed an adaptive learning engine, E-gostky that relies on Vygotsky's Zone of Proximal Development. E-gostky uses machine learning models to skip content in the learning curriculum by dynamically assessing the learning potential of the student at the current time. It was evaluated in a large-scale field experiment and compared to a baseline path which was built by domain experts. Results show that students using E-gostky maintained comparable performance while spending 17% less time, guessed less and reported higher engagement and motivation.

9. ACKNOWLEDGMENTS

We thank the Center for Educational Technology (CET) in Tel-Aviv for their contribution to this study.

10. ADDITIONAL AUTHORS

Additional authors: Galit Lukin (MIT, email: glukin@mit.edu) and Daniel Sitton (Microsoft, email: dsitton@microsoft.com).

11. REFERENCES

- [1] R. S. Baker, A. T. Corbett, K. R. Koedinger, and A. Z. Wagner. Off-task behavior in the cognitive tutor classroom: when students game the system. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 383–390. ACM, 2004.
- [2] J. Beck, M. Stern, and B. P. Woolf. Using the student model to control problem difficulty. In *User Modeling*, pages 277–288. Springer, 1997.
- [3] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [4] S. Chaiklin. The zone of proximal development in vygotsky's analysis of learning and instruction. *Vygotsky's educational theory in cultural context*, 1:39–64, 2003.
- [5] L. Darling-Hammond. No child left behind and high school reform. *Harvard Educational Review*, 76(4):642–667, 2006.
- [6] Y. B. David, A. Segal, and Y. K. Gal. Sequencing educational content in classrooms using bayesian knowledge tracing. In *Proceedings of the sixth international conference on Learning Analytics & Knowledge*, pages 354–363. ACM, 2016.
- [7] E. Joseph. Engagement tracing: using response times to model student disengagement. *Artificial intelligence in education: Supporting learning through intelligent and socially informed technology*, 125:88, 2005.
- [8] J. P. Lantolf and M. E. Poehner. Dynamic assessment in the classroom: Vygotskian praxis for second language development. *Language Teaching Research*, 15(1):11–33, 2011.
- [9] P. M. Moreno-Marcos, P. J. Muñoz-Merino, C. Alario-Hoyos, I. Estévez-Ayres, and C. D. Kloos. Analysing the predictive power for anticipating assignment grades in a massive open online course. *Behaviour & Information Technology*, 37(10-11):1021–1036, 2018.
- [10] R. Pelánek. Exploring the utility of response times and wrong answers for adaptive learning. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, page 18. ACM, 2018.
- [11] A. Segal, Y. B. David, J. J. Williams, K. Gal, and Y. Shalom. Combining difficulty ranking with multi-armed bandits to sequence educational content. In *International Conference on Artificial Intelligence in Education*, pages 317–321. Springer, 2018.
- [12] C. Tieso. The effects of grouping practices and curricular adjustments on achievement. *Journal for the Education of the Gifted*, 29(1):60–89, 2005.
- [13] O. Vainas, O. Bar-Ilan, R. Gilad-Bachrach, G. Lukin, M. Ronen, R. Shillo, and D. Sitton. E-gostky: Sequencing content using the zone of proximal development. *arXiv preprint arXiv:1904.12268*, 2019.
- [14] L. S. Vygotsky. *Mind in society: The development of higher psychological processes*. Harvard university press, 1980.
- [15] S. Walpole and M. C. McKenna. *Differentiated reading instruction: Strategies for the primary grades*. Guilford Press, 2007.

Hǎo Fāyīn: Developing Automated Audio Assessment Tools for a Chinese Language Course

Ashvini Varatharaj
Worcester Polytechnic Institute
100, Institute Rd
Worcester, MA
avaratharaj@wpi.edu

Anthony F. Botelho
Worcester Polytechnic Institute
100, Institute Rd
Worcester, MA
abotelho@wpi.edu

Xiwen Lu
Worcester Polytechnic Institute
100, Institute Rd
Worcester, MA
oluxiwen@gmail.com

Neil Heffernan
Worcester Polytechnic Institute
100, Institute Rd
Worcester, MA
nth@wpi.edu

ABSTRACT

We present and evaluate a machine learning based system that automatically grades audios of students speaking a foreign language. The use of automated systems to aid the assessment of student performance holds great promise in augmenting the teacher's ability to provide meaningful feedback and instruction to students. Teachers spend a significant amount of time grading student work and the use of these tools can save teachers a significant amount of time on their grading. This additional time could be used to give personalized attention to each student. Significant prior research has focused on the grading of closed-form problems, open-ended essays and textual content. However, little research has focused on audio content that is much more prevalent in the language-study education. In this paper, we explore the development of automated assessment tools for audio responses in a college-level Chinese language-learning course. We analyze several challenges faced while working with data of this type as well as the generation and extraction of features for the purpose of building machine learning models to aid in the assessment of student language learning.

Keywords

Audio Analysis, Automatic Grading, Machine Learning

1. INTRODUCTION

Learning proper pronunciation is an essential aspect of learning to speak a new language. Almost all standardized language tests involve a section where the person being evaluated is expected to speak out loud; these verbal tests are used to assess student skill and knowledge of pronunciation, fluency, and the correct usage of vocabulary. In the previous years, computer-assisted pronunciation teaching (CAPT) has

gained attention and has been commercialized such as Pearson, SRI, and ETS [6]. Language learning is a common part of many educational systems, and language classes often involve assignments which are related to speaking tasks. This gives us, as researchers and developers of tools to aid in the teacher assessment of students, an opportunity to collect and utilise language students' audio responses. The audio data of language learners are rich data-sets which provide insights about how well students are acquiring language skills based on the pronunciations and quality of speech. These data-sets may also identify problem areas where a student may be in need of improvement. However, collecting this data can be challenging because the data is not commonly stored uniformly and all in one place. Since oral tests are given in class, recording and storing this data would add an overhead to the teacher's responsibilities. Additionally, this process can occur more efficiently. The students are all evaluated for on similar measures like the standardized exams that test pronunciation and fluency. By automating the process of grading students, we can both help learners self-evaluate their progress and provide tools to teachers who traditionally grade students by listening to audio files (a task that can be very time consuming considering the number of potential students a teacher may have in a single class). The grading systems which have been researched previously are usually for more closed-form responses. Open ended responses, such as essays or explanatory, answers are a more challenging task to automatically grade, but there has been growing research on developing automated assessment tools for such tasks [5].

This paper presents an exploratory analysis representing an initial step toward developing automated assessment tools for language learning audio responses. In this paper, we explore the development of automated graders of student audio responses from a Chinese language class. We seek to address the following research questions: 1. By employing models of varying complexity, are we able to automatically grade student audio responses better than a simple majority class baseline model? 2. Does a recurrent deep learning model outperform a static decision tree model in regard in predicting student grades? 3. Which features extracted from student audio responses provide the greatest impact on model performance in predicting student grades?

Ashvini Varatharaj, Anthony Botelho, Xiwen Lu and Neil Heffernan "Hao Fa Yin: Developing Automated Audio Assessment Tools for a Chinese Language Course" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 663 - 666

2. RELATED WORK

Automatic speech recognition systems have been the focus of several works over the past three decades. Some of this notable research has been conducted on the use of Automatic Speech Recognizer (ASR) technology for automatic speech scoring and the evaluation of pronunciation quality (cf. [7]). With Deep Learning improving the performance of many tasks, it has been widely used in the Automatic Speech Recognition and Scoring tasks [4]. Many states have begun to use computer-based English Learning Proficiency (ELP) assessments, which has led to a growing interest in automated scoring of spoken responses to increase the efficiency of scoring. However, there is a dearth of systems which grades foreign language learners. In this paper we plan to focus on Chinese learning Undergraduate students.

3. DATASET

The data-set is collected from a Chinese language class for undergraduate students in a university located in the North-East region of the United States. The data collected consists of 63 distinct students from 3 classes run between 2017 and 2019. All of these classes were taught by the same instructor. The data is comprised of 60 audio recordings which includes audio responses from 3 different prompts. The average length of the audio is approximately 2.5 minutes. All work is submitted through an institute-hosted learning management system from which the teacher then downloads all the files in order to listen and grade each student's work. In our data-set, the teacher followed a rubric when assessing each student that is reported in Table 1

3.1 Pre-processing

A series of pre-processing steps were applied to the raw audio responses for use in this work. Among the 60 responses, in 12 cases, responses included multiple students speaking; it is likely that permission was given to these students to work collaboratively on the assignment. In 2 of these cases, a separate grade was given to each student present in the recording while in the remaining 9 cases, the students involved were given the same score. We observed that the grades in these 2 cases where the score differed did not vary more than 1 point (on a scale of 11, from 0-10 inclusively), and therefore we aggregated each of the responses and grades into a single instance by averaging the two scores. The other 9 cases were left as individual samples, leaving us with the aforementioned 60 distinct responses for use in our models. For the feature extraction step (described in the next section), we further need to convert our audio data in a mono-channel format. The data we collected contains stereo data (i.e. it contains a separated right and a left channel). To convert each response to a mono signal, we took the average of the two channels of the stereo data.

Grading Components	Percentage
Rich Content	40%
Grammar and word usage	20%
Accuracy of tones and pronunciation	20%
Fluency	20%

Table 1: The grading rubric used to assess student audio responses.

4. METHODOLOGY

Our methodology includes feature extraction and building models for predicting teacher-provided grades for student audio responses. We compared two non-linear models to a simple baseline. While the grade labels were provided on an 11-point scale (0-10), this scaling is non-linear due to non-uniformity across the grades (the average grade was 7.41). Since few of the assignments were graded on different scales, a min-max scaling was used to transform all the scores into a scale of 0-10.

4.1 Feature Extraction

Audio feature extraction is performed to transform the audio signals recorded in the .wav files into a representation which can be used for machine learning. The features used are extracted using the *PyAudioAnalysis* library; this is a python-based library which is exclusively used for audio data feature extraction. We use the default parameters of 50 milliseconds and 25 milliseconds for the window size and step amount respectively. With these parameters, the feature extraction function split the input signal into short windows (frames), leading to a sequence of short-term feature vectors at regular intervals within the signal. The number of windows varied for each of the signals based on the length of the audio file. The features extracted for use in this work are briefly described in Table 2. For each time window, we extracted the 34 features, where several of the features described in Table 2 are described using a multi-valued numeric vector indicated through the Feature ID column.

4.2 Deep Learning Model

The sequential and temporal aspects of audio data makes the application of a recurrent deep learning model an appropriate choice for developing automated assessment tools. Specifically, we utilize a Long Short Term Memory (LSTM) [3] network, as it is designed to model complex temporal relationships within sequential data. This model observes a sequence of time steps (e.g. frames of audio as described in the Feature Extraction Section) and is trained to produce a single value corresponding to the estimated grade for the student response. Due to the number of features and length of each student response, we chose a network structure with 3 hidden layers. The input of the model is represented as a sequence of 34-valued vectors corresponding with the extracted features, which is then passed to a LSTM hidden layer of 50 nodes, before being passed through 2 additional fully connected non-recurrent layers of 100 units each. An output layer of a single node is used corresponding with the grade of the student, treated as a regression rather than a classification task. We chose this structure to prevent the network from overfitting due to the long sequences (i.e. by using a smaller LSTM layer), but providing enough depth in the model to learn feature representations from each sequence; exploring additional model structures is planned for future work. The LSTM looks at each frame and provides an estimated grade for it, but is only updated and evaluated on the final frame of the sequence. We applied a 5-fold cross validation on the data-set and measure performance using RMSE and Spearman correlation.

4.3 Decision Tree Model

To contrast the deep learning network, we also compare a decision tree model which has the capacity to learn non-linear

Feature ID	Feature Name	Description
1	Zero Crossing Rate	The rate of sign-changes of the signal during the duration of a particular frame.
2	Energy	The sum of squares of the signal values, normalized by the respective frame length.
3	Entropy of Energy	The entropy of sub-frames' normalized energies. It can be interpreted as a measure of abrupt changes.
4	Spectral Centroid	The center of gravity of the spectrum.
5	Spectral Spread	The second central moment of the spectrum.
6	Spectral Entropy	Entropy of the normalized spectral energies for a set of sub-frames.
7	Spectral Flux	The squared difference between the normalized magnitudes of the spectra of the two successive frames.
8	Spectral Rolloff	The frequency below which 90% of the magnitude distribution of the spectrum is concentrated.
9-21	MFCCs	Mel Frequency Cepstral Coefficients form a cepstral representation where the frequency bands are not linear but distributed according to the mel-scale.
22-33	Chroma Vector	A 12-element representation of the spectral energy where the bins represent the 12 equal-tempered pitch classes of western-type music (semitone spacing).
34	Chroma Deviation	The standard deviation of the 12 chroma coefficients.

Table 2: The 34 Features extracted from the audio along with its description.

relationships between the features and teacher-provided grades, but does so in a non-sequential manner. As such, we needed to aggregate the audio features into a single vector that describes the response as a whole as input to the model. We took the average across each of the 34 features across each response and used it to predict the teacher-provided grade. We applied a decision tree regressor using the CART algorithm [1]. Similar to the deep learning model, we evaluated the model using a 5-fold cross validation using measures of RMSE and Spearman correlation. Within each training fold we optimized it for its depth using the training data.

4.4 Baseline Method

We compare each the LSTM model and decision tree model to a simple baseline using a majority class model. For this method, we take the average grade provided by the teacher and use this as a prediction for every sample. It can be used to evaluate how well our models perform in comparison to a model that incorporates no audio information.

5. RESULTS

We report model performance using measures of RMSE, a measure of prediction error, and Spearman correlation (Rho). The performance of each of our models in predicting the audio response grades is reported in Table 3. From this, it can be seen that both models outperform the baseline model in regard to RMSE, but only the LSTM model exhibited positive correlation. These results demonstrate that the LSTM is the superior model, although the values suggest that there is still room for improvement. As the grade labels follow a 10 point scale, the best RMSE of 2.728 exhibited by the LSTM suggests that it is, on average, over- or under-predicting the true grade of the student by just under 3 grade points. Despite this room for improvement, the results do suggest that both the LSTM and decision tree models are learning from the data in potentially different ways. We further explore each of these models through an ablation study.

6. ABLATION STUDY

In this experiment, we perform an ablation study where we run a model with all the features and iteratively remove

Model	RMSE	Rho
Majority Class	3.323	-
Decision Tree	2.807	-0.076
LSTM	2.728	0.163

Table 3: Audio Grade prediction: average 5-fold RMSE and R2 score for the models

each to observe impacts to model performance. Changes in model performance as a result of removing a feature can be used then as a measure of feature importance in determining the grade of the student. Table 4 reports the results of this study across both the LSTM and Decision Tree models. The rows in the table are sorted to reflect the features of highest impact found in regard to changes in RMSE for the LSTM model as this was the highest performing model across both metrics.

In regard to the decision tree model, the 3 features which cause the largest drop in RMSE are Energy of the wave, MFCC features, and Spectral Centroid. With respect to the Spearman correlation metric, the top three correlated features are the Chroma features followed by the Zero Crossing Rate and MFCC features. In the LSTM model, the 3 features which cause the largest decrease in the RMSE are the 12 Chroma features, the Energy Entropy, and the Zero Crossing Rate feature. However, comparing the Spearman's correlation measure (rho) does not follow the same trend as the RMSE. In the case of LSTM, most of the models have a rho value more than the model with all the features. This may suggest overfitting within the model, particularly as some of the features similarly lead to improvements in RMSE when removed.

7. DISCUSSION

As can be seen from the decision tree model as well as the LSTM model, the energy related features has a significant impact on the evaluation of the pronunciation. In [2] it was shown that 'formants' are bands of energy around a particular frequency which characterizes different resonances of the vocal tract and it helps understand pronunciation of vow-

Feature Removed	LSTM		Decision Tree	
	Delta RMSE	Rho	Delta RMSE	Rho
Chroma	0.118	0.172	0.217	0.162
Entropy of Energy	0.076	0.18	0.143	0.074
Zero Crossing Rate	0.042	0.149	0.132	0.121
Spectral Centroid	0.042	0.148	0.226	0.072
Spectral Spread	0.04	0.189	0.205	0.085
Spectral Rolloff	0.028	0.206	0.136	0.107
Spectral Entropy	0.025	0.217	0.132	0.121
Chroma Deviation	0.018	0.208	0.059	0.076
Energy	-0.011	0.19	0.361	-0.025
Spectral Flux	-0.016	0.242	0.151	0.11
MFCC	-0.18	0.161	0.314	0.132

Table 4: Ablation Study results from the LSTM and Decision Tree model

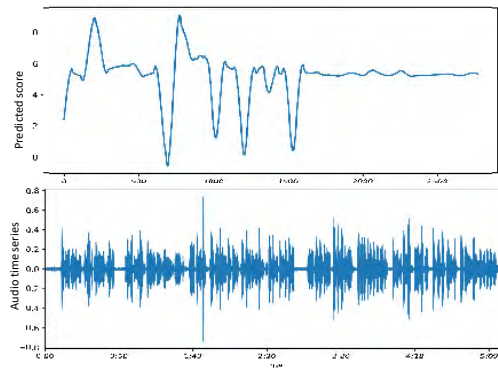


Figure 1: The Waveform from an audio of a student answer along with the predictions by the LSTM model at each window.

els. The second feature that has an impact in the decision tree is the MFCC feature. The MFCC features accurately characterizes the envelope of the short time power spectrum which manifests the shape of the vocal tract. Hence it makes sense that it influences the pronunciation score. However, the LSTM model seems to understand MFCCs differently and hence, the removal of these MFCC features seem to having a improvement in the model. Further analysis on the MFCC features can help us understand why this is the case. The Chroma feature provides information related to the 12 musical octaves. Both the LSTM and the decision tree model seem to show an increase in the RMSE when these features are removed.

A benefit of the sequential structure of the LSTM model is its ability to illustrate the development of its grading estimates over the audio response. From moment-to-moment, a grade estimate can help to indicate sections of the audio response that suggest a high grade (e.g. well-pronounced words) and sections that suggest a low grade (e.g. poor pronunciation or areas of silence); an example of this is illustrated in Figure 1. In this figure, the bottom image depicts the wave form of a student audio response while the top figure illustrates the LSTM estimate over the length of the response. Such a report could help teachers to identify sections of audio where the student may be in need of additional aid.

8. CONCLUSION AND FUTURE WORK

Based on the results we plan to follow the following steps for the future. First, exploring additional model architectures may lead to more accurate assessment tools. Second, we plan on incorporating contextual knowledge since 40 percent of the grade includes content. Converting the audio to text and extracting text-related features could potentially provide more understanding of the evaluation of the audio. And finally, using LSTMs the scores for each segment of audio can be graded and we plan on using it to aid students in self-assessment and to help teachers learn where their students need further aid. This work is an initial step toward the development of automated assessment tools designed to aid language learning students and teachers. We hope that further in-depth analyses of different combinations of the features will better help understand these relationships.

9. REFERENCES

- [1] L. Breiman. *Classification and regression trees*. Routledge, 2017.
- [2] V. Fridland, K. Bartlett, and R. Kreuz. Do you hear what i hear? experimental measurement of the perceptual salience of acoustically manipulated vowel variants by southern speakers in memphis, tn. *Language variation and change*, 16(1):1–16, 2004.
- [3] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [4] K. Kyriakopoulos, K. M. Knill, and M. J. Gales. A deep learning approach to assessing non-native pronunciation of english using phone distances. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, volume 2018, pages 1626–1630, 2018.
- [5] K. Taghipour and H. T. Ng. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891, 2016.
- [6] S. M. Witt. Automatic error detection in pronunciation training: Where we are and where we need to go. *Proc. IS ADEPT*, 6, 2012.
- [7] K. Zechner, D. Higgins, X. Xi, and D. M. Williamson. Automatic scoring of non-native spontaneous speech in tests of spoken english. *Speech Communication*, 51(10):883–895, 2009.

A Meta-Learning Augmented Bidirectional Transformer Model for Automatic Short Answer Grading

Zichao Wang
Rice University
Houston, TX 77005
jzwang@rice.edu

Andrew S. Lan
University of Massachusetts
Amherst
Amherst, MA 01003
andrewlan@cs.umass.edu

Andrew E. Waters
OpenStax
Houston, TX 77005
aew2@rice.edu

Phillip Grimaldi
OpenStax
Houston, TX 77005
phillip.grimaldi@rice.edu

Richard G. Baraniuk
Rice University & OpenStax
Houston, TX 77005
richb@rice.edu

ABSTRACT

We introduce ml-BERT, an effective machine learning method for automatic short answer grading when training data, i.e., graded answers, is limited. Our method combines BERT (Bidirectional Representation of the Transformer), the state-of-the-art model for learning textual data representations, with meta-learning, a training framework that leverages additional data and learning tasks to improve model performance when labeled data is limited. Our intuition is to use meta-learning to help us learn an *initialization* of the BERT parameters in a specific target subject domain using unlabeled data, thus fully leveraging the limited labeled training data for the grading task. Experiments on a real-world student answer dataset demonstrate the promise of ml-BERT method for automatic short answer grading.

1. INTRODUCTION

We consider the problem of automatic grading for short answer questions that require students to provide concise *textual* responses. Figure 1 shows an example of short answer questions. The pedagogical benefits of these questions over multiple-choice questions have been studied and validated by [11]. Unfortunately, manually grading short answer questions is labor intensive, rendering it extremely challenging to administer these questions in large-scale educational settings.

In this paper, we aim to advance the state-of-the-art in automatic short answer grading. This problem can be viewed as a supervised (machine) learning problem where we would like to grade (classify) answers as correct or incorrect given a dataset of answers and their grades (labels).¹ A method capable of reliably grading short answer questions gives instructors more freedom in choosing the form of assessments in large-scale educational settings where

¹We will use “grade” and “label” interchangeably throughout the paper.

Question:	What is a difference between saturated and unsaturated fats?
Answer:	Saturated fats have no double bonds whereas unsaturated fats have at least one double bond

Figure 1: A sample question and correct student answer in their raw textual form taken from a high school biology class.

manual grading is unfeasible. In practice, however, the number of graded answers is very limited since manual grading is labor-intensive. This constraint makes training a classifier for automatic grading a highly challenging task. Thus, it is desirable to develop a method that can effectively leverage the ungraded answers to learn a representation of the answers and achieve satisfactory performance with only a few graded answers. This feature is valuable for instructors in large-scale educational settings because they would need to manually grade only a few answers, which saves them time and effort that are better allocated to other pedagogical actions. Therefore, in this work, we focus on this particular scenario where we only have access to a limited number of graded answers.

A number of prior works have demonstrated the effectiveness of machine learning methods in automatic grading [2, 8, 9, 10]. As a high-level summary, a typical method first converts textual data composed of questions and answers to some vector representations and then uses them to classify an answer as correct or incorrect. In our problem setting, however, this method for automatic short answer grading faces two major challenges. First, we need a *model* capable of producing high quality representations of textual data that captures the information in both the question text and the answer text, which is essential for the classifier to make its decision. We note that learning a representation of textual data has been a challenge not only for short answer grading but also for the broader field of natural language processing (NLP). Second, since the number of graded answers is limited, we need a specialized *training procedure* that enables the classifier to remain effective using only a limited number of labels. We note that this procedure is important because a number of prior works, e.g., [4], have demonstrated that regular training procedures may result in poor results when labeled data is limited. For example, we have merely less than 20k labeled answers whereas related classification tasks such as sentiment analysis have over 1 million labeled examples [5], which contributes to the success of machine learning models on those tasks.

Zichao Wang, Andrew Lan, Andrew Waters, Phillip Grimaldi and Richard Baraniuk "A Meta-Learning Approach to Automatic Short Answer Grading" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 667 - 670

To tackle the first challenge and learn textual data representations efficiently and effectively, we resort to BERT (bidirectional encoder representation of the transformer), the state-of-the-art model for text. Since BERT achieves the best results to date in a wide range of benchmark NLP tasks, we develop our method on top of it. To tackle the second challenge and fight the scarcity of labeled data, we resort to *meta-learning*, an emerging training procedure that searches for a good initialization of the model parameters by using unlabeled data to optimize additional objectives that are unrelated to the target task. Previous studies [4, 7] have shown that parameter initialization is critical for the success of neural network models including BERT; Thus, the meta-learning procedure helps us to find a better initialization of the model parameters and quickly adapt to new tasks with only limited labeled data.

1.1 Contributions

We propose ml-BERT, a meta-learning method that augments the bidirectional encoder representation of the transformer (BERT) model for automatic short answer grading with limited labeled answers. Our method consists of two learning phases. First, in the meta-learning phase, we optimize for an initialization of BERT parameters using unlabeled data. We carefully choose data from a specific educational domain to use in the meta-learning phase such that it is closely related to the questions in the short answer grading task. This choice ensures that the meta-learned initialization builds a good representation of the domain and is highly useful to the subsequent short answer grading task. Second, in the regular learning phase, we optimize BERT parameters for the short answer grading task using limited labeled training data, starting from the meta-learned parameter initialization in the previous phase. Experimental results on a real-world dataset consisting of student answers to a set of questions in high school biology shows that the proposed ml-BERT method is more effective than regular BERT and several other baseline methods when labeled data is limited.

The rest of the paper is organized as follows. Section 2 details our ml-BERT method. Section 3 presents experimental results on a real-world student answer dataset and discusses the key findings. Finally, Section 4 summarizes our work and proposes future research directions. This paper is a truncated version; for more details including background on BERT and meta-learning, exhaustive literature review and additional experimental results, please refer to the long version of this paper on ArXiv with the same title.

2. METHOD

In this section, we describe ml-BERT, our novel training procedure that augments BERT with meta-learning for short answer grading. The ml-BERT training procedure consists of two learning phases, namely, the *meta-learning phase* and the *regular learning phase*. In the meta-learning phase, we optimize for an initialization of BERT parameters that ultimately leads to better short answer grading performance. In the regular learning phase, we optimize for the BERT parameters using labeled short answer dataset to further improve the model performance on short answer grading.

Key to the success of ml-BERT is the choice of datasets and tasks in the meta-learning phase for learning parameter initialization. We choose *language modeling* and *next sentence prediction* as our meta-learning tasks. We use a dataset that includes textbooks, question texts, and correct student answer texts for the language modeling task and a dataset that includes textbooks for the next sentence prediction task. The specific choice of the above data depends on the educational domain. For example, we would choose relevant

biology textbooks if the task is to grade answers to biology related questions. Intuitively, the language modeling task helps us learn an initialization that captures the nuances of the language usage specific to each educational domain. The next sentence prediction task helps us learn an initialization that captures the logic flow underlying the textbooks from which questions are asked. Initialization learned from the above tasks and data is thus informative to the subsequent short answer grading task because it contains some knowledge of whether an answer uses appropriate language and whether an answer is logically related to its associated question and the specific educational domain.

Below, we first formally define the target task which is short answer grading. We then describe the two meta-learning tasks and their datasets. Finally, we present the full ml-BERT training procedure.

Target task: short answer grading. This is a supervised learning problem that we explicitly train the model to perform well on. The training dataset for short answer grading consists of N examples $\{q_i, a_i, y_i\}_i^N$, where q_i and a_i are text segments representing question and student short answer, respectively, and y_i is a binary variable indicating whether the answer is correct or incorrect. The learning objective is to correctly classify each answer given the question. We use the negative log-likelihood loss to measure this objective:

$$\mathcal{L}_T = -\frac{1}{N} \sum_{i=1}^N \log p(y_i | a_i, q_i) \quad (1)$$

where $p(y_i | a_i, q_i)$ is modeled by a composition of functions including BERT mapping $f^{(1)}(\cdot)$ of the first “[CLS]” token (see Section 2.1 of the long version of this paper),² a fully connected layer $g : \mathbb{R}^M \rightarrow \mathbb{R}$, and a sigmoid function $\sigma(\cdot)$:

$$p(y_i | a_i, q_i) = \sigma \left(g \left(f^{(1)}(a_i, q_i) \right) \right).$$

In practice, in order to obtain a single textual input for BERT, we append all tokens of each answer a_i to all tokens of its associated question q_i , append “[CLS]” and “[SEP]” respectively at the beginning and end of the concatenated token sequence, and separate the answer and question tokens with “[SEP]”.

Meta-learning task #1: language modeling. This is an unsupervised learning problem. The dataset $\mathcal{D}_1 = \{s_i\}_{i=1}^{N_1}$ consists of a collection of N_1 sentences s_i taken from text corpora specific to the educational subject. Each sentence is represented by a sequence of J_i tokens: $s_i = \{w_j\}_{j=1}^{J_i}$ (see Section 2.1 of the long version of this paper).

During training, we randomly replace a portion of all tokens in each sentence with the special token “[MASK]” and ask the model to predict these masked tokens. We set the portion of masked tokens to 15% for each sentence. Performance of token prediction is measured by negative log-likelihood loss \mathcal{L}_1 as:

$$\mathcal{L}_1 = -\frac{1}{N_1} \sum_{i=1}^{N_1} \sum_{k=1}^{J_i} \log p(w_{ik} | w_1, \dots, w_{ik-1}, w_{ik+1}, \dots, w_{J_i}) \quad (2)$$

where w_{ik} represents the k^{th} masked token in the i^{th} sentence. The conditional probability distribution is modeled using the composition of BERT mapping $f(\cdot)$, a linear layer $g_i : \mathbb{R}^M \rightarrow \mathbb{R}^V$

²In BERT, we obtain the sentence representation by querying the embedding of this “[CLS]” token.

Table 1: Statistics of the dataset for each task. Note that the short answer grading dataset consists of only labeled answers which represents less than 10% of all available answers.

tasks	#examples	#average tokens
Short answer grading	495 questions 18143 answers	17.88 per question 12.31 per answer
Masked language modeling	78684 sentences	19.08 per sentence
Next sentence prediction	21052 sentences	25.54 per sentence

that maps every BERT encoded token into a vector of dimension V which is the size of the vocabulary, and a softmax function:

$$p(w_k | w_1, \dots, w_{k-1}, w_{k+1}, \dots, w_J) = \text{softmax}(g_1(f(\tilde{s})))$$

where we drop the example index i for notation simplicity. \tilde{s} denotes the input sentence with the k^{th} token replaced by the mask which indicates which token the model should predict.

We note that there are other strategies for language modeling with BERT. Conventionally, one models language by predicting the next token in the input sentence given all previous tokens [6]. We choose this *masked* prediction strategy for language modeling because of its superior empirical performance than the conventional approach [3].

Meta-learning Task #2: next sentence prediction. This is an unsupervised learning problem. The dataset $\mathcal{D}_2 = \{s_i, \hat{s}_i, z_i\}_{i=1}^{N_2}$ consists of N_2 pairs of sentences sampled from textbook chapters specific to the educational subject of the short-answer questions. We sample the sentence \hat{s}_i such that half of the time it is the next sentence of s_i and half of the time it is a sentence from a random location in the textbook. z_i is a binary variable indicating whether \hat{s}_i is the next sentence of s_i . We use negative log likelihood \mathcal{L}_2 to measure model performance on next sentence prediction:

$$\mathcal{L}_2 = -\frac{1}{N_2} \sum_{i=1}^{N_2} \log p(z_i | s_i, \hat{s}_i) \quad (3)$$

where the conditional probability is modeled by the composition of BERT mapping $f^{(1)}(\cdot)$ of the first “[CLS]” token, a fully connected layer $g_2: \mathbb{R}^M \rightarrow \mathbb{R}$, and a sigmoid function $\sigma(\cdot)$:

$$p(y_i | s_i, \hat{s}_i) = \sigma \left(g_2 \left(f^{(1)}(s_i, \hat{s}_i) \right) \right).$$

The ml-BERT training procedure. Model parameter update according to ml-BERT proceeds as follows. In the meta-learning phase, we update the BERT parameters by alternating between the two tasks until we reach a pre-specified stopping condition. In the regular learning phase, we initialize the BERT parameters from the meta-learned parameter initializations in the previous phase and update the parameters using the labeled short answer dataset until we reach a pre-specified stopping condition. In both phases, we perform parameter update using gradient descent optimizers. The exact choice of the optimizer is flexible; we use the customized Adam optimizer for BERT outlined in [3]. The proposed ml-BERT procedure is simple to implement and effective in practice, which we demonstrate in the next section.

3. EXPERIMENTAL RESULTS

We now demonstrate the effectiveness of our method using real-world data. We first introduce various model and training settings

and then explain our results in detail. Code for our experiments can be shared upon request.

Dataset and pre-processing steps. We collect real-world short answers from semester-long biology classes that use the OpenStax Biology textbook.³ Table 1 summarizes the key statistics of the dataset. Notably, only about 10% of the answers are graded; we use only the graded responses as training data. Heaping the power of the vast number of ungraded responses is left for future work. We perform an 80/20 training/validation split independently for each question. We also discard questions in the training set with less than 20 labeled answers, 10 for each label, and questions in the validation set with less than 4 labeled answers, 2 for each label. This is to ensure we have well-balanced training and validation splits of the dataset. For the language modeling task during meta-learning, we use all textbook text, questions, instructor-authored answers associated with the OpenStax Biology textbook. For the next sentence prediction task, we only use the textbook text.

The textual data is minimally pre-processed. We first turn all texts to lowercase. For BERT, we use the WordPiece tokenizer to process the input text into a sequence of tokens. For other baseline models, we use a bag-of-words (BoW) representation of textual data, which involves a more complicated pre-processing pipeline including tokenization, lemmatization and stop word removal.

Hyper-parameters. We adapt the standard BERT model configuration and training setup. We use BERT’s customized Adam optimizer with a learning rate of 3×10^{-5} . Training lasts 10 epochs with a batch size of 32. Each question-answer pair is limited to 128 word pieces; shorter ones are padded and longer ones are truncated. Most of the setup information is available in the official BERT code release.⁴

Baseline models. We compare ml-BERT with four baseline methods: 1) baseline BERT without meta-learning, 2) Random Forest, 3) K-Nearest Neighbors and 4) logistic regression. The baseline BERT model has the same hyper-parameter setup as BERT with meta-learning. We use default settings in the python `sklearn` package for the remaining three models. In addition, logistic regression is the best linear classifier among those that we tested; therefore we use logistic regression as the representative linear classifier baseline.⁵

3.1 Quantitative Analysis

Table 2 compare the classification accuracy and F1 score of ml-BERT to that of the four baseline models on the validation set. We clearly see ml-BERT improves both classification accuracy and F1 score compared to the baseline models. Note in particular that BERT without meta-learning only achieves comparable performance with random forest, but with meta-learning it is able to outperform all baseline models. This suggest that in the limited labeled answer scenario, meta-learning has the potential to lift the model performance further.

To get a fine-grained performance analysis, We compare the performance of ml-BERT and BERT on questions of different levels according to Bloom Taxonomy. Bloom Taxonomy [1] categorizes

³<https://openstax.org/details/books/biology>

⁴<https://github.com/google-research/bert>

⁵Recall that K-Nearest Neighbors and Random Forest are nonlinear classifiers.

Table 2: Evaluation accuracy and F1 score comparing ml-BERT with various baselines.

Models	eval. acc.	eval. F1
Logistic Regression	71.39%	0.723
K Nearest Neighbors	72.74%	0.735
Random Forest	77.82%	0.768
Baseline BERT	77.80%	0.788
ml-BERT	80.17%	0.815

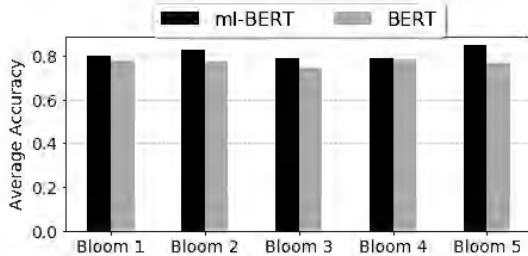


Figure 2: Grading accuracy per Bloom level comparing ml-BERT to BERT. We see improved accuracy across questions with different Bloom’s levels.

questions into 6 distinct levels where higher level indicates more cognitive load required to answer a question. We can thus use Bloom levels as rough difficulty measure of the questions. Figure 2 summarizes the classification accuracy for each bloom level. We see that ml-BERT outperforms BERT in grading questions across all bloom levels, most significantly at Bloom level 5. This can be explained by the benefit of meta-learning which learns language patterns and logical relations in the specific educational subject that helps grade these “difficult” questions with higher accuracy.

3.2 Influence of Each Meta-Learning Task

We investigate the effect of each of the meta-learning tasks on the quality of the learned representations of text. Specifically, we compare ml-BERT using both learning tasks with ml-BERT using only one of the learning tasks. We summarize the comparison in Table 3. Results suggest that, overall, both tasks contribute to the quality of the learned representations. We emphasize that, even with only one of the meta-learning tasks, ml-BERT is able to adapt to the short answer grading task and achieve better performance compared to baseline BERT. This observation highlights the importance of using task-specific data and tasks in the meta-learning phase. Moreover, it also suggests that ml-BERT has potential in further improving its performance as we incorporate more tasks specific to the target educational domain, which we leave for future work.

4. CONCLUSIONS AND FUTURE WORK

We have introduced ml-BERT, a method to augment the state-of-the-art text embedding model BERT with meta-learning to improve its performance on automatic short answer grading. In the first phase of ml-BERT, we use meta-learning to learn an initialization of BERT parameters by training language modeling and next sentence prediction tasks on data specific to the relevant educational domain. In the second phase, we further optimize the model parameters using limited labels on the correctness of the short answers. We experimentally validate the effectiveness of ml-BERT on real-world student answers collected in high school biology classes. Both quantitative and qualitative results demonstrate that the proposed method is promising.

Table 3: The impact of each of the two meta-learning tasks on grading accuracy. We see that the best results are achieved when we use both meta-learning tasks. We also observe that using either one of the meta-learning tasks already leads to improvement over baseline BERT.

Conditions	eval. acc.	eval. F1
ml-BERT		
- Without masked language modeling	79.12%	0.805
- Without next sentence prediction	78.63%	0.793

5. ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their constructive feedback and Shashank Sonkar for helpful discussions. This work was partially supported by Bill and Melinda Gates Foundation, Arthur & Carlyse Ciocca Charitable Foundation, NSF grants IIS-17-30574 and IIS-18-38177, and DOD Vannevar Bush Faculty Fellowship (NSSEFF) grant N00014-18-1-2047.

6. REFERENCES

- [1] P. Armstrong. *Bloom’s Taxonomy*, 2014 (accessed Mar. 4, 2019).
- [2] S. Burrows, I. Gurevych, and B. Stein. The eras and trends of automatic short answer grading. *Int. J. Artificial Intell. in Edu.*, 25(1):60–117, Mar. 2015.
- [3] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv e-prints*, 1810.04805, Oct 2018.
- [4] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta learning for fast adaptation of deep networks. In *Proc. Int. Conf. Mach. Learn.*, volume 70, pages 1126–1135, Aug. 2017.
- [5] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report*, 2009.
- [6] D. Jurafsky and J. H. Martin. *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., 2009.
- [7] Q. V. Le, N. Jaitly, and G. E. Hinton. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. *ArXiv e-prints*, 1504.00941, Apr. 2015.
- [8] M. Mohler and R. Mihalcea. Text-to-text semantic similarity for automatic short answer grading. In *Proc. Conf. Eur. Chapter Assoc. Comput. Linguistics*, pages 567–575, Mar. 2009.
- [9] M. A. Sultan, C. Salazar, and T. Sumner. Fast and easy short answer grading with high accuracy. In *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Language Technol.*, pages 1070–1075, Jun. 2016.
- [10] N. Suzen, A. Gorban, J. Levesley, and E. Mirkes. Automatic Short Answer Grading and Feedback Using Text Mining Methods. *arXiv e-prints*, 1807.10543, Jul. 2018.
- [11] A. Waters, P. Grimaldi, A. S. Lan, and R. G. Baraniuk. Short-answer responses to stem exercises: Measuring response validity and its impact on learning. In *Proc. Conf. Edu. Data Mining*, pages 374–375, Jun. 2017.

Deep Hierarchical Knowledge Tracing

Tianqi Wang
University at Buffalo
Buffalo, NY
twang47@buffalo.edu

Fenglong Ma
University at Buffalo
Buffalo, NY
fenglong@buffalo.edu

Jing Gao
University at Buffalo
Buffalo, NY
jing@buffalo.edu

ABSTRACT

Knowledge tracing is an essential and challenging task in intelligent tutoring systems, whose goal is to estimate students' knowledge state based on their responses to questions. Although many models for knowledge tracing task are developed, most of them depend on either concepts or items as input and ignore the hierarchical structure of items, which provides valuable information for the prediction of student learning results. In this paper, we propose a novel deep hierarchical knowledge tracing (DHKT) model exploiting the hierarchical structure of items. In the proposed DHKT model, the hierarchical relations between concepts and items are modeled by the hinge loss on the inner product between the learned concept embeddings and item embeddings. Then the learned embeddings are fed into a neural network to model the learning process of students, which is used to make predictions. The prediction loss and the hinge loss are minimized simultaneously during training process.

Keywords

knowledge tracing, hierarchical structure modeling, deep learning

1. INTRODUCTION

Knowledge tracing is an essential and challenging task in intelligent tutoring systems. The goal of knowledge tracing task is to estimate the mastery state of a specific knowledge component based on students' responses to items. In other words, knowledge tracing aims to predict the correctness of a student's response to the next item according to all the previous response records.

In order for a student to answer an item correctly, he/she needs to master the concepts related to this item first. For example, a student can provide correct responses to both "1 + 1" and "28 + 36", which illustrates that this student may master the general concept of addition. Some existing knowledge tracing models [1, 8, 9] are proposed to predict the students' performance only based on general concept information of items. A common drawback of such models is

that they ignore the differences among different items even under the same general concept. In fact, if a student knows how to solve the items related to the concept "Addition of Two Integers", this student may correctly answer "28 + 36" but make a mistake when answering a harder item "285 + 361". In addition, the learning gains of answering different items related to the same concept are different. Correctly solving a more complex item indicates a higher gain towards the desired knowledge states than that obtained by solving an easier one. To distinguish and model the differences among items, Item Response Theory model [10] is proposed, which directly uses items as the input to estimate a student's ability. However, students may visit these online platforms very infrequently and only attempt on a small subset of items. Therefore, for each item in the dataset, only a small number of attempts are made, which leads to the issue of data sparsity. On such sparse data, existing knowledge tracing models, for example, Item Response Theory [10], that takes items as input may have limited performance. Deep knowledge tracing [9], which applies RNN to predict the performance of students, has shown improved prediction performance in knowledge tracing, but it requires a large amount of data for training. Such models would suffer more from the data sparsity issue.

To handle the data sparsity issue and better distinguish items, we propose a novel deep hierarchical knowledge tracing (DHKT) model, which can leverage the hierarchical information between items and concepts. Specially, DHKT learns the embeddings of items and concepts and models the relations among items and concepts by calculating the hinge loss of the inner product of the embeddings. The main contribution of this work can be summarized as follows: We propose a novel DHKT model by leveraging the hierarchical structure between items and concepts into the state-of-the-art deep knowledge tracing model. Experimental results show that the DHKT model learns meaningful representations and outperforms the state-of-the-art baselines.

2. METHODOLOGY

In this section, we first introduce how to model the students' learning process using a deep learning framework, and then illustrate how to incorporate the concept-item graph into the model.

2.1 Problem Formulation

We denote the set of students as \mathcal{K} . For a student $k \in \mathcal{K}$ interacting with the system t times, the interactions are denoted as $\mathcal{X}_k = \{x_{k,1}, x_{k,2}, \dots, x_{k,t}\}$. In this work, the inter-

Tianqi Wang, Fenglong Ma and Jing Gao "Deep Hierarchical Knowledge Tracing" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 671 - 674

actions specifically refer to the students' responses to corresponding items. $x_{k,t} = (i_{k,t}, y_{k,t})$ is a tuple representing the item $i_{k,t} \in \mathcal{I}$ attempted by student k at time t , where \mathcal{I} represents the set of all items, and $y_{k,t} \in \{0, 1\}$ represents the correctness of the response. $y_{k,t} = 0$ indicates an incorrect response and $y_{k,t} = 1$ represents a correct one.

These items are related to different knowledge concepts, which are the more general representations of the items. The set of knowledge concepts is denoted as \mathcal{C} in this work. The relations between items and concepts are annotated by experts. We denote the mapping matrix, i.e., the concept-item graph, as $\mathcal{Q} \in \{0, 1\}^{|\mathcal{C}| \times |\mathcal{I}|}$. If $q_{m,n} = 1$, which means the item n is related to the concept m ; otherwise, $q_{m,n} = 0$.

With the above definitions, the knowledge tracing task can be formulated as a binary sequence prediction problem: Given a series of interactions $\mathcal{X}_k = \{x_{k,1}, x_{k,2}, \dots, x_{k,t}\}$, $x_{k,t} = (i_{k,t}, y_{k,t})$ of a student k , the next item $i_{k,t+1} \in \mathcal{I}$ and the concept-item mapping matrix \mathcal{Q} , our goal is to predict the value of $y_{k,t+1}$ representing if student k will answer the new given item $i_{k,t+1}$ correctly based on the current knowledge state $\mathbf{h}_{k,t}$ of the student.

2.2 Model Student Learning

We use dense embeddings instead of one-hot encoding as the input of the DHKT model. These dense embeddings can be automatically learned from the training dataset. To distinguish the difference among items related to the same concept and preserve the concept-level information, the concatenation of item embedding and concept level embedding is employed to represent an item in the DHKT model.

Let $\mathbf{e}_{i_{k,t}} \in \mathbb{R}^d$ denote the embedding of the item $i_{k,t}$, where d is the dimension of the embedding. Since one item $i_{k,t}$ can be related to more than one concept, we use the average embeddings of all the concepts related to $i_{k,t}$ as its concept-level embedding $\mathbf{e}_{c_{k,t}}$. Mathematically, the concept-level embedding for the item $i_{k,t}$ is defined as:

$$\mathbf{e}_{c_{k,t}} = \frac{1}{|\mathcal{C}_{k,t}|} \sum_{c_m \in \mathcal{C}_{k,t}} \mathbf{e}_{c_m}, \quad (1)$$

where $\mathcal{C}_{k,t}$ denotes the set of concepts related to item $i_{k,t}$, $|\mathcal{C}_{k,t}|$ is the number of such concepts, c_m represents a concept in $\mathcal{C}_{k,t}$, and \mathbf{e}_{c_m} is the embedding of the concept c_m .

The new hierarchical representation of the item $i_{k,t}$ can be described as the concatenation of item and concept embeddings:

$$\mathbf{v}_{k,t} = \mathbf{e}_{i_{k,t}} \oplus \mathbf{e}_{c_{k,t}}, \quad (2)$$

where $\mathbf{v}_{k,t} \in \mathbb{R}^{2d}$ and \oplus denotes vector concatenation.

To jointly represent the item and the correctness of student k 's response, we introduce $\mathbf{a}_{k,t} \in \mathbb{R}^{4d}$ as:

$$\mathbf{a}_{k,t} = \begin{cases} \mathbf{v}_{k,t} \oplus \mathbf{0} & y_{k,t} = 1 \\ \mathbf{0} \oplus \mathbf{v}_{k,t} & y_{k,t} = 0 \end{cases}, \quad (3)$$

where $\mathbf{0} \in \{0\}^{2d}$ is the zero-vector. $\mathbf{a}_{k,t}$ is the input when we model the process of student learning.

In the student learning process, the current knowledge state of a student is highly correlated with the previous knowledge state and the learning gains from the new materials. Thus, the student learning process can be modeled by Long Short-

Term Memory network [3] as:

$$\begin{aligned} \mathbf{g}_{k,t} &= \sigma(\mathbf{W}_i \mathbf{a}_{k,t} + \mathbf{U}_i \mathbf{h}_{k,t-1} + \mathbf{b}_i), \\ \mathbf{f}_{k,t} &= \sigma(\mathbf{W}_f \mathbf{a}_{k,t} + \mathbf{U}_f \mathbf{h}_{k,t-1} + \mathbf{b}_f), \\ \mathbf{o}_{k,t} &= \sigma(\mathbf{W}_o \mathbf{a}_{k,t} + \mathbf{U}_o \mathbf{h}_{k,t-1} + \mathbf{b}_o), \\ \mathbf{r}_{k,t} &= \mathbf{f}_{k,t} \otimes \mathbf{r}_{k,t-1} \\ &\quad + \mathbf{g}_{k,t} \otimes \tanh(\mathbf{W}_c \mathbf{a}_{k,t} + \mathbf{U}_c \mathbf{h}_{k,t-1} + \mathbf{b}_c), \\ \mathbf{h}_{k,t} &= \mathbf{o}_{k,t} \otimes \tanh(\mathbf{r}_{k,t}), \end{aligned} \quad (4)$$

where h denotes the dimensionality of hidden state vector. $\mathbf{g}_{k,t}, \mathbf{f}_{k,t}, \mathbf{o}_{k,t}, \mathbf{r}_{k,t}, \mathbf{h}_{k,t} \in \mathbb{R}^h$ are the activation vector of the input gate, forget gate, output gate, the memory cell and the hidden state vector of student k at time t . $\mathbf{W}_i, \mathbf{W}_f, \mathbf{W}_o, \mathbf{W}_c \in \mathbb{R}^{h \times 2d}$ and $\mathbf{U}_i, \mathbf{U}_f, \mathbf{U}_o, \mathbf{U}_c \in \mathbb{R}^{h \times h}$ are weight matrices, $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_o, \mathbf{b}_c \in \mathbb{R}^h$ is the bias vector which need to be learned during training. \otimes denotes element-wise product. The $\sigma(\cdot)$ and $\tanh(\cdot)$ denote the Sigmoid and Hyperbolic Tangent function separately.

The correctness of a student's response to an item is dependent on both the current knowledge state of the student and the characteristics of the item. Thus we use the concatenation of student k 's current knowledge state $\mathbf{h}_{k,t}$ outputted by the LSTM and the representation of item $i_{k,t+1}$ that denotes the characteristics of the item, i.e., $\mathbf{v}_{k,t+1}$, to make prediction. The concatenated vector is fed into a fully connected layer to obtain a summary vector $\mathbf{s}_{k,t}$, and then this vector is fed into a Sigmoid activation layer to calculate the probability of correctly answering item $i_{k,t+1}$ by student k . The process can be represented as:

$$\begin{aligned} \mathbf{s}_{k,t} &= \tanh(\mathbf{W}_{fc}(\mathbf{h}_{k,t} \oplus \mathbf{v}_{k,t+1}) + \mathbf{b}_{fc}), \\ p_{k,t+1} &= \sigma(\mathbf{W}_s \mathbf{s}_{k,t} + \mathbf{b}_s), \end{aligned} \quad (5)$$

where $\mathbf{W}_{fc} \in \mathbb{R}^{d_s \times (h+2d)}$, $\mathbf{W}_s \in \mathbb{R}^{d_s}$, $\mathbf{b}_{fc} \in \mathbb{R}^{d_s}$ and $\mathbf{b}_s \in \mathbb{R}^1$ are the weight matrices and biases to be learned, and d_s denotes the dimension of the summary vector. $p_{k,t+1}$ is the probability that student k can answer item $k+1$ correctly.

2.3 Hierarchical Structure Constraint

In fact, there exists a concept-item graph between items and concepts. The concept-item graph provides us with the grouping information of the items. The items related to the same concept can be considered as belonging to the same group. They should be similar with other items within the same group, while dissimilar with items in other groups. At the same time, the concept should capture the characteristics of all items related to it and can approximately represent all these items.

Based on the above analysis, we introduce a hinge loss which tries to maximize the margins among different groups to model the hierarchical structure of items. We apply the embeddings of concepts to represent the general group characteristics of all the items related to the concept. When deriving the representations of items and concepts, we keep the item similar to its corresponding concepts, and on the contrary, make it far away from other concepts. Thus, the hinge loss between an item n and a concept m is defined as

$$l_h^{m,n} = \begin{cases} \max\{0, 1 - \mathbf{e}_{i_n}^T \mathbf{e}_{c_m}\} & q_{m,n} = 1 \\ \max\{0, 1 + \mathbf{e}_{i_n}^T \mathbf{e}_{c_m}\} & q_{m,n} = 0 \end{cases}, \quad (6)$$

where \mathbf{e}_{i_n} and \mathbf{e}_{c_m} are the embeddings of the item i_n and concept c_m , and $q_{m,n}$ is an indicator in the item to concept

mapping matrix Q indicating whether the item n is related to concept m . $q_{m,n} = 1$ indicates that item n is related to concept m , while $q_{m,n} = 0$ means that item n is not related to concept m .

2.4 Loss Function

The objectives of the proposed model are two folds: One is to make accurate predictions of students' responses, and the other is to learn meaningful embeddings of items. Based on these two objectives, in the training stage, we need to minimize the prediction loss and the hinge loss simultaneously.

For student k answering item $i_{k,t}$ at time t , the prediction loss $l_{k,t}$ can be modeled with the binary cross-entropy:

$$l_{k,t} = -(y_{k,t} \log(p_{k,t}) + (1 - y_{k,t}) \log(1 - p_{k,t})), \quad (7)$$

where $p_{k,t}$ is the probability that student k can answer item $i_{k,t}$ correctly and $y_{k,t}$ is the correctness of the response of student k at time t .

The total loss can be represented as the weighted sum of the total prediction loss and the total hinge loss:

$$\mathcal{L} = \sum_{k=1}^{|\mathcal{K}|} \sum_{t=1}^{t_k} l_{k,t} + \alpha \sum_{m=1}^{|\mathcal{C}|} \sum_{n=1}^{|\mathcal{I}|} l_h^{m,n}, \quad (8)$$

where t_k is the total number of questions that a student k attempted, and $|\mathcal{K}|$ is the number of students. $|\mathcal{I}|$ and $|\mathcal{C}|$ are the number of items and concepts in \mathcal{I} and \mathcal{C} respectively. $l_h^{m,n}$ is the hinge loss defined in Eq. 6. α is a hyper-parameter to balance the weight of prediction loss and hinge loss.

3. EXPERIMENTS

In this section, we present the experiments that evaluate the proposed DHKT model on knowledge tracing task. These experiments are performed on three real-world datasets.

3.1 Datasets

The ASSIST09¹ and ASSIST12² datasets were collected from the ASSISTments tutoring system [2]. In the experiment, we use skill builder dataset. In the preprocessing, we remove all duplicated records and the records without a skill id or without a skill name and the records without a $\{0, 1\}$ value of the "correctness" attribute. In addition, we also remove the students with a sequence length less than three. After preprocessing, there are 3,991 students, 227,156 records, 13,876 items and 96 concepts in the ASSIST09 dataset. The ASSIST12 dataset contains 270,66 students, 2,541,201 records, 45,716 items and 245 concepts after preprocessing.

The Statics dataset³ was collected from a college-level engineering statics course [5]. This dataset includes transactions from two different modes: tutor mode and assessment mode. Since the transactions from assessment mode for the same student have the same timestamp, we cannot determine the order of these transactions. Thus, only the transactions from tutor mode are included in the experiment. Also, we remove the students with less than three transactions. After preprocessing, there are 317 students, 137,711 records, 987 items

¹<https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data>

²<https://sites.google.com/site/assistmentsdata/home/2012-13-school-data-with-affect>

³<https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=507>

Table 1: Overview of the Three Datasets.

	ASSIST09	ASSIST12	Statics
# of students	3,991	27,066	317
# of items	13,876	45,716	987
# of concepts	96	245	280
# of records	227,156	2,541,201	137,711
attempts per student	57	94	434
items per concept	145	187	4
attempts per item	16	56	140
attempts per concept	2,366	10,372	492

and 280 concepts. The statistics of the three datasets are shown in Table 1.

3.2 Baselines and Experimental Settings

To evaluate the effectiveness of the DHKT model, we compare the DHKT model with the state-of-the-art knowledge tracing models, including Item Response Theory (IRT) [10], Hierarchical Item Response Theory (HIRT) [10], Performance Factor Analysis (PFA) [8], Bayesian Knowledge Tracing (BKT) [1] and Deep Knowledge Tracing (DKT) [9]. IRT, HIRT, which is a Bayesian extension of IRT by considering the hierarchical structures between concepts and items, and PFA make predictions based on the logit function. BKT models the sequence of responses and makes predictions based on the Hidden Markov Model. DKT applies RNN to model the response sequence and make predictions. To evaluate the effect of incorporating concept-item graph in deep knowledge tracing, the variations of the proposed DHKT model: EDKT, Fine-grained EDKT and DHKT-, are also compared. These variations share the same network structure with DHKT, but they are different in terms of the input and the value of α . EDKT and Fine-grained EDKT use the item embedding and the concept embedding as the input separately and $\alpha = 0$. DHKT- is the reduced model of DHKT where only the item embedding is used as input in Eq. (2). In the ASSIST datasets the skill_id is considered as the concept and the problem_id is considered as the item. In the Statics dataset, the problem_name is the concept and the step_name is the item. The PFA, BKT, DKT and EDKT take concepts as input while IRT, Fine-grained EDKT and DHKT- take items as input. The concept-item graphs are constructed according to the relations between items and concepts and are used by HIRT and DHKT.

We split each of these datasets into training and testing datasets on student level. For each dataset, we randomly select 20% of the students as the testing dataset and keep 80% of the students as the training dataset to learn the parameters. We randomly select 20% of the training students for validation. The training and testing datasets for all the models are the same. For training the DHKT model, batch size is set to 32, and the number of epochs is set to 100. The hidden state dimensionality h is set to 100. The hyper-parameters are tuned on the validation datasets. We tune the embedding dimensionality and the balance parameter α using grid search. The candidate values for embedding dimensionality d are $\{25, 50, 100\}$, and α 's in Eq. (8) are $\{0.001, 0.01, 0.1, 1\}$. The loss function is optimized by Adam algorithm [4], which is a gradient-based optimization algorithm based on adaptive estimates of lower-order moments. We set the learning rate to 0.01. To avoid the exploding gradient problem, gradient norm clipping strategy [7] is

Table 2: AUC Values on the Three Datasets.

Model	ASSIST09	ASSIST12	Statics
IRT	0.6891	0.7317	0.8249
HIRT	0.6912	0.7234	0.8251
PFA	0.7040	0.6706	0.7705
BKT	0.6722	0.6141	0.7318
DKT	0.7483	0.7346	0.7736
EDKT	0.7513	0.7310	0.7823
Fine-grained EDKT	0.7342	0.7428	0.8251
DHKT-	0.7780	0.7677	0.8311
DHKT	0.7866	0.7747	0.8333

adopted at the threshold of 20. We use dropout with a probability of 0.6 to alleviate the overfitting issue.

The Area Under ROC Curve (AUC) is used to evaluate the performance of all the models, in which ROC curve plots true positive rate versus false positive rate in a binary classification task. For each model, we run five times with random initialization and report the average AUC. The AUC of the testing dataset is calculated using the model with the highest validation AUC value among 100 epochs.

3.3 Results

The AUC values for different models on all three datasets are shown in Table 2. The proposed DHKT model achieves the highest AUC on all the three datasets. The reduced model DHKT- cannot beat DHKT, but it still outperforms all other baselines on the three datasets. The improvement from Fine-grained EDKT to DHKT- demonstrates the effectiveness of incorporating the concept-item graph. The difference in the performance between DHKT and DHKT- indicates that exploiting general level information is useful in deep knowledge tracing.

3.4 Embedding Visualization

We use t-SNE [6] to visualize the learned embeddings of items by DHKT in a 2-D space to qualitatively assess the interpretability of the representations. For comparison, we also plot the learned item embeddings on the three datasets of the Fine-grained EDKT. The color of the dots represents the concept related to the items.

The learned embeddings are shown in Figure 1. Figure 11(a), 11(b) and 11(c) show the learned item representations of DHKT on ASSIST09, ASSIST12 and Statics, and Figure 11(d), 11(e) and 11(f) show the learned item representations of the Fine-grained EDKT model on corresponding datasets. Compared with the items mixed together learned by Fine-grained EDKT, the item embeddings learned by DHKT are well separated and more consistent with the hierarchical structures on the ASSIST09 and ASSIST12 datasets. On the Statics dataset, although some clusters are mixed with each other, the representations learned by DHKT are much better than that learned by Fine-grained EDKT. In addition, the prediction performance of DHKT is better than that of Fine-grained EDKT on the three datasets, which demonstrates the importance of meaningful item representations for knowledge tracing.

4. CONCLUSIONS AND FUTURE WORK

In this work, we propose a novel deep hierarchical knowledge tracing model by incorporating the hierarchical structure of items. The proposed model not only improves the performance of knowledge tracing task, but also provides

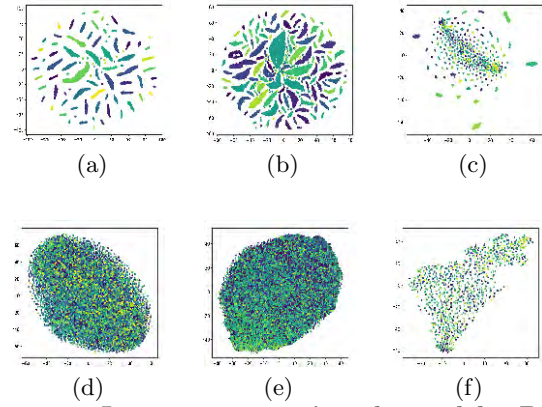


Figure 1: Item representations learned by DHKT and Fine-grained EDKT.

meaningful representations of items. The item representations learned by the proposed model are consistent with the hierarchical structure of the items. The superior prediction performance indicates that the hierarchical structure of items plays an important role in deep knowledge tracing, and meaningful representations can help improve deep knowledge tracing performance.

We plan to investigate how to apply multi-level hierarchical structures in knowledge tracing and how to recommend learning materials and items to students based on their knowledge state in the future.

5. ACKNOWLEDGEMENTS

This work is sponsored by NSF IIS-1553411. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

6. REFERENCES

- [1] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [2] M. Feng, N. Heffernan, and K. Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3):243–266, 2009.
- [3] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [4] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [5] K. R. Koedinger, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the edm community: The psic datashop.
- [6] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.
- [7] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.
- [8] P. I. Pavlik Jr, H. Cen, and K. R. Koedinger. Performance factors analysis—a new alternative to knowledge tracing. *Online Submission*, 2009.
- [9] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.
- [10] K. H. Wilson, Y. Karklin, B. Han, and C. Ekanadham. Back to the basics: Bayesian extensions of irt outperform neural networks for proficiency estimation. *arXiv preprint arXiv:1604.02336*, 2016.

Beyond Autoscoring: Extracting Conceptual Connections from Essays for Classroom Instruction

Korah J. Wiley, Allison Bradford, Zachary Pardos, Marcia C. Linn
University of California, Berkeley
Berkeley, CA 94720-1670
{korah.wiley, allison_bradford, pardos, mclinn}@berkeley.edu

ABSTRACT

While automated essay evaluation techniques have dramatically reduced instructors' grading burden, they fall short of providing instructors with the rich qualitative insights into students' sense making process that a careful read of essays can afford. In this study, we demonstrate how word embedding techniques can serve as a complement to automated scoring, providing instructors with valuable near-time insight into how their students are conceptualizing targeted lesson concepts. For this study, we use a post-test essay associated with two Web-based Inquiry Science Environment (WISE) units that provides instruction about how the sun causes increases in temperature. We create word2vec models fit to students' c-rater scored essay responses at each score level of a rubric designed to assess students' integrated understanding of targeted concepts. Using cosine similarity, we identify, with statistically reliability, the ideas that students at each score level of the rubric used in relation to the concepts targeted in the essay prompt. Our instructor interview reveals the validity of the results in providing insight into students' ideas, differentiated across understanding levels.

1. INTRODUCTION

In the domain of science, new science education reform efforts, like the Next Generation Science Standards (NGSS), call for students to both coherently understand and communicate complex science ideas[6]. Consequently, essays that assess students' developing knowledge of complex science ideas could increase the validity of assessment in science classrooms[6]. Concomitant with the call for increased use of essay assessments is the need for machine-based techniques to quickly and reliably analyze student essays in order to provide instructors with qualitative insights about how their students are developing and connecting complex science concepts.

Advances in the field of natural language processing (NLP) have given rise to automated essay evaluation (AEE) tech-

niques that help instructors meet the challenges of essay scoring. However, there is still the need to develop effective techniques to assess and support students' comprehension of complex ideas expressed in their essays[9]. In order for instructors to provide targeted support based on their students' developing ideas, they need to have the qualitative insight that comes from reading the essays. Without it, instructors are left in the dark regarding the different ways their students make sense of the targeted concepts being assessed by the essay item. Even though instructors may have access to the exemplars used to train AEE models, the distance between the exemplar responses and that of their students can leave instructors guessing about the true nature and quality of their students' understanding. In this study, we describe the development and evaluation of word2vec models that augment the value of automated scoring by analyzing and comparing the conceptual connections that students in different scoring categories express in their essay.

2. BACKGROUND

Research in the field of teaching and learning has shown students' ability to develop an integrated understanding of complex ideas, such energy transfer and transformation, is influenced by how well their instructors notice and understand their ideas[8]. Therefore, to support students in communicating integrated understanding of complex ideas, it is important to develop machine-based analyses that provide teachers opportunities to see, understand, and respond to student ideas.

2.1 Auto-scored Student Essays

In partnership with Educational Testing Services (ETS), we have previously used the c-rater algorithm to score student essays from various assessment items in Web-based Inquiry Science Environment (WISE) units[2]. The student responses used to train the c-rater model were human-coded using a rubric based on the Knowledge Integration (KI) framework[4]. The KI framework supports students to sort through their ideas to develop an integrated understanding of normative science concepts[3]. Since the rubric used to score the essay assessment items in the units prioritizes the links that students make between normative science ideas, the c-rater generated scores reflect the extent to which students' ideas are normative and linked.

To support instruction based on students' ideas, we create skip-gram models[5] of students' c-rater-scored essay re-

Korah Wiley, Allison Bradford, Zach Pardos and Marcia Linn
"Beyond Autoscoring: Extracting Conceptual Connections from Essays for Classroom Instruction" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 675 - 678

sponses. In doing so, we were able to augment automated essay feedback and, thus, provide instructors with quantitative information regarding the correctness of students' ideas and qualitative information regarding the connectedness of their ideas.

2.2 Science Content Knowledge

The WISE units used in this study were designed to support students in developing an integrated understanding of how energy from the sun is transferred and transformed and how those processes cause an increase in temperature. The assumption is that students at each score level conceptualize and connect these science ideas differently. Since the word2vec models encode the semantic meaning of the words as vectors, we can, for example, analyze the way students scoring at the level of 2 conceptualize "temperature" as compared to how students at a level 5 conceptualize "temperature". Using cosine similarity, we can provide teachers with a list of the words that their students have conceptually connected to the target lesson concept (e.g. temperature).

Given this potential, we conducted this study to address the following research questions:

1. Can we develop a skip-gram model from pre-scored essay responses to reliably extract the differential ways students develop and conceptually connect complex science ideas?
2. Do the skip-gram models generate results that are pedagogically informative (i.e. support teachers to notice, understanding, and respond to their students ideas)?

3. METHODOLOGY

3.1 Offline Model Development (RQ1)

Student essay responses were prepared for skip-gram development using standard Python libraries, pywsd and NLTK were used to perform basic NLP techniques (i.e. abbreviation expansion, case-adjustment, punctuation and symbol exclusion, lemmatization, and stopword modification and removal)[1]. Skip-gram models for student responses at the KI rubric score levels (range 2-5, four total) were developed using the gensim library[7]. Model hyperparameters were adjusted to produce cosine similarity results to the various target words that a content-expert validated as conceptually relevant to the prompt. To extract the meaning of a word as used and understood by the student rather than its conventional meaning, we created the word vectors from the study datasets rather than using pre-trained vectors.

3.2 Classroom Evaluation (RQ2)

To determine the efficacy and utility of our skip-gram models to support teacher noticing of student ideas, we conducted a semi-structured interview of the high-school physics instructor whose students' essay responses generated the model results referenced during the interview.

3.3 Datasets

The dataset used in the offline development and evaluation of the skip-gram models consisted of student text responses to the explanation portion of the following post-test essay assessment item:

Let's think about how global warming happens. On a COLD day, Akbar walks to his car that is parked in the sun and has not been driven for a week. Predict the temperature inside the car:

- Colder than the outside air
- Warmer than the outside air
- Exactly the same as the outside air

Explain your answer:

The students of four sixth grade science teachers from two Bay Area middle schools (N=497) generated their responses to this prompt after engaging the WISE Global Climate Change unit (<https://wise.berkeley.edu/project/24751>). In this unit, students learned how solar radiation from the sun can be absorbed, transformed into heat energy, and trapped by greenhouse gases as infrared radiation, which leads to increased temperature.

Student responses were, on average, 34 words long and were human-coded, from 1 (low) to 5 (high), using a KI rubric. Responses received a score of 1 if they were off-task or irrelevant, and, thus, were excluded from the model development dataset. Responses that included normative but unconnected ideas about the transformation of solar radiation to heat received a score of 3. Responses that connected one or more normative ideas about the transformation of solar radiation to heat received a score of 4 or 5, respectively. The distribution of scores were as follows: Score 2 = 243; Score 3 = 92; Score 4 = 42; Score 5 = 23.

The dataset used in the classroom evaluation of the skip-gram models consisted of student text responses to the same essay prompt used for the offline model development and evaluation (see above). The students of a ninth grade physics instructor from a Bay Area high school generated their responses (N=155) before and after engaging the WISE Solar Ovens unit, which focused on designing, building, and testing a solar oven (<https://wise.berkeley.edu/project/24537>). In this unit, students learned about the same energy cycle described in the sixth grade unit, and then explored how different designs influence that energy cycle and temperature change in a solar oven.

Student responses were, on average, 30 words long and were scored using a c-rater algorithm based on the same KI rubric used for the model development dataset. Similarly, responses received a score of 1 if they were off-task or irrelevant and were excluded from the dataset. The distribution of scores were as follows: Score 2 = 82; Score 3 = 46; Score 4 = 22; Score 5 = 3. Due to sparseness of responses at the score level 5, a skip-gram model was not created for this score level.

4. RESULTS

4.1 Offline Model Development (RQ1)

We chose "heat" and "temperature" as target words for our model evaluation, since the essay used in this study was designed to assess students understanding of how solar radiation and infrared radiation cause an increase in temperature. Our initial cosine similarity results from the four skip-gram models appeared to be consistent with our content expert's

Table 1: The cosine similarity results for the target word "temperature" of the skip-gram models associated with each score level using the development dataset. The vocabulary size of each model is in parentheses.

Temperature							
Score = 2 (105)	cos_sim	Score = 3 (55)	cos_sim	Score = 4 (62)	cos_sim	Score = 5 (58)	cos_sim
become	0.999	get	0.998	inside	0.985	get	0.987
reason	0.999	into	0.994	get	0.985	glass	0.981
colder	0.999	become	0.987	glass	0.981	inside	0.981
inside	0.999	glass	0.986	into	0.969	become	0.976
time	0.996	inside	0.977	become	0.968	into	0.961
get	0.995	particle	0.967	travel	0.967	car	0.935
hot	0.995	may	0.933	radiation	0.955	bounce	0.906
warmer	0.993	bounce	0.932	bounce	0.922	pass	0.87

Table 2: The most similar words and associated p-values for the target word "heat" of the skip-gram models associated with each score level using the development dataset. The vocabulary size of each model is in parentheses.

Heat							
Score = 2 (105)	p-value	Score = 3 (55)	p-value	Score = 4 (62)	p-value	Score = 5 (58)	p-value
outside	<1.0E-6	outside	2.42E-04	car	1.13E-04	car	0.009
inside	<1.0E-6	inside	0.012	air	0.007	glass	0.009
temperature	<1.0E-6	air	0.049	glass	0.045	infrared	0.009
cold	<1.0E-6	energy	0.049	radiation	0.045	energy	0.042
air	<1.0E-6	get	0.049	turn	0.045	day	0.042
warm	<1.0E-6	into	0.049			light	0.042
get	3.58E-06	warm	0.049			solar	0.042
sun	0.039	trap	0.049				

expectations regarding the ways that students are different scoring levels would connect ideas related to "temperature" (see Table 1). However, these initial results were not reproducible from one model run to the next, in terms of exact words and relative cosine similarity rank. The top 8 results for each model within a given run displayed patterns consistent with the content expert's expectations, albeit with variable reproducibility.

We used distribution probability to establish the statistical reliability of our model results. Specifically, we used hypergeometric and binomial distribution test, respectively, to determine the probability that any given word in the model's vocabulary would appear in the top 8 cosine similarity results and do so consistently enough to yield a p-value < 0.05. We ran each model 12 consecutive times on a random sample of the essay responses, where the number of responses in the sample equaled the total number of responses in the dataset.

Using the development dataset, we generated a statistically reliable list of the words based on the model results for the target word "heat" (see Table 2). Examination of this list revealed conceptually meaningful differences across the scoring levels, as confirmed by the content expert and physics instructor. These model results indicate a typical progression of student ideas when they are beginning to understand the mechanism of how solar radiation transforms to heat.

4.2 Classroom Evaluation (RQ2)

To investigate the alignment of the model results with the instructor's expectations of the conceptual connections that

Table 3: The most similar words for the target word "temperature" of the skip-gram models associated with each score level using the classroom dataset. The vocabulary size of each model is in parentheses.

Temperature		
Score = 2 (92)	Score = 3 (68)	Score = 4 (62)
become	get	inside
reason	into	get
colder	become	glass
inside	glass	into
time	inside	become
get	particle	travel
hot	may	radiation
warmer	bounce	bounce

students at each score level would make, we presented the instructor with the cosine similarity results from each model for the target word "temperature" (see Table 3). Although the statistical reliability for the model results had not yet been established, we asked the instructor to interpret the results for each individual model and cross-comparatively. The instructor commented that the results from the model resonated with her expectations of what a developed understanding looks like and what a still-developing understanding looks like.

Furthermore, we asked the instructor when and how she might use the information provided by our models. She indicated that she would use a tool like this as a formative assessment to see what her students' ideas were while instruc-

tion was still ongoing. She went on to say that she could use the information from the models to inform her instruction, and that it would help her to decide if she needed to have a whole class conversation, do a demo, or show a video to support her students in sorting and integrating their ideas towards normative understandings.

5. DISCUSSION

5.1 Typical Classroom Datasets

The ability to support teaching and learning by developing unsupervised models using essay response data has historically been stymied by the need to train these models on large datasets (i.e. hundreds of responses)[9]. Since our aim was to provide instructors with specific, rather than generalizable insight into student thinking, we were able to use a small dataset with as few as 23 responses in a scoring category to train our skip-gram models. Our models demonstrate that the data generated from one instructor's teaching load (i.e. 4 class periods of 30-40 students) may be sufficient to provide meaningful insight into student thinking. Therefore, using their small-scale data, instructors with relatively small class sizes (e.g. K-12 teachers, liberal arts instructors, or seminar leaders) can benefit from the insight afforded by word embedding models.

By training the models for specific instructor-student contexts rather than for general use across all instructors, we can give instructors insight into how their group of students are understanding the target lesson concepts. This subpopulation-specific insight affords instructors the ability to tailor their instruction based on the ideas held by their students rather than on the population-level conceptions provided by large-scale research. Armed with such nuanced insight, instructors can feasibly engage in knowledge integration-based instruction in which they attend to and build on students' ideas.

5.2 Capitalizing on Essay Assessments

Education reform efforts that call for students to communicate their understanding of complex ideas have identified essays as valid and effective learning assessments[6]. The results from this study highlight the ability of word embedding models to help instructors capitalize on the affordances that essays have for revealing how students conceptualize and connect complex ideas.

The automatically generated c-rater scores allow the teacher to know which students hold at least one normative idea (score 3) and which can make normative conceptual links (scores 4 and 5). However, this does not provide them with information about what these ideas are or of the other potentially productive, but non-normative ideas their students are holding. The conceptual connections shown by our models give an indication of the ideas, both normative and non-normative, that students are holding as they answer the essay question. Therefore, the output from our models augments the instructional value of autoscored essays by providing instructors with statistically reliable, qualitative insight with which to see, understand, and respond to student ideas. Since the models are developed from essays at each score level, upon viewing the results, an instructor can identify a potential learning progression and how to support their

students in further developing an integrated understanding of the complex ideas.

6. CONCLUSION AND NEXT STEPS

The results from this study provide clear next steps toward supporting teaching and learning. First, we need to determine the most effective means of displaying the model results and develop an interface through which instructors can view the results and explore various conceptual connections. Our partner instructors have expressed that their confidence in computer model results is linked to the extent to which they understand how the model generates the results. Therefore, it is our research priority to develop a model that both generates meaningful results and can be easily explained. Second, the utility of the results in supporting student learning needs to be further validated by evaluating student outcomes based on targeted instructor interventions.

Although the essays used in this study were from the science domain, there is broad application for the technique to other domains, as the only prerequisite is that the text responses be precategorized, either by scoring or other means.

7. REFERENCES

- [1] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001-. [Online; accessed 07 September 2018].
- [2] C. Leacock and M. Chodorow. Crater: Automated Scoring of Short-Answer Questions. *Language Resources and Evaluation - LRE*, 37:389–405, 2003.
- [3] M. C. Linn and B.-S. Eylon. *Science learning and instruction: taking advantage of technology to promote knowledge integration*. Routledge, New York, 2011. OCLC: ocn176946367.
- [4] O. L. Liu, J. A. Rios, M. Heilman, L. Gerard, and M. C. Linn. Validation of automated scoring of science assessments. *Journal of Research in Science Teaching*, 53(2):215–233, Feb. 2016.
- [5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA, 2013. Curran Associates Inc.
- [6] National Research Council. *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas*. The National Academies Press, Washington, DC, 2012.
- [7] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. <http://is.muni.cz/publication/884893/en>.
- [8] A. D. Robertson, R. E. Scherr, and D. Hammer, editors. *Responsive teaching in science and mathematics*. Teaching and learning in science series. Routledge, Taylor & Francis Group, New York, 2016.
- [9] K. Zupanc and Z. Bosnic. Advances in the field of automated essay evaluation. 39:383–395, 2015.

What You Say is Relevant to How You Make Friends: Measuring the Effect of Content on Social Connection

Yiqiao Xu

Department of Computer
Science
NCSU, Raleigh, NC, USA
yxu35@ncsu.edu

Collin F. Lynch

Department of Computer
Science
NCSU, Raleigh, NC, USA
cflynch@ncsu.edu

Niki Gitinabard

Department of Computer
Science
NCSU, Raleigh, NC, USA
ngitina@ncsu.edu

Tiffany Barnes

Department of Computer
Science
NCSU, Raleigh, NC, USA
tmbarnes@ncsu.edu

ABSTRACT

Discussion forums are the primary channel for social interaction and knowledge sharing in Massive Open Online Courses (MOOCs). Many researchers have analyzed social connections on MOOC discussion forums. However, to the best of our knowledge, there is little research that distinguishes between the types of connections students make based upon the content of their forum posts. We analyze this effect by distinguishing on- and off-topic posts and comparing their respective social networks. We then analyze how these types of posts and their social connections can be used to predict the students' final course performance. Pursuant to this work we developed a binary classifier to identify on- and off-topic posts and applied our analysis with the hand-coded and predicted labels. We conclude that the post type does affect the relationship between the students and their closest neighbors or community members clustered communities and their closest neighbor to their learning outcomes.

Keywords

MOOC, social network analysis, forum participation

1. INTRODUCTION & BACKGROUND

Social interactions are an essential component of learning. Peer collaborators in courses provide support by engaging in informal advising, sharing "institutional knowledge", and engaging in the co-construction of knowledge [6]. Students who lack strong social connections are more prone to feeling lost or discouraged in a course and are more likely to drop out [12]. This issue is of particular interest in Massive Open Online Courses (MOOCs), which seek to scale classroom instruction to hundreds or even thousands of students supported by a single instructor. Prior researchers

have shown that students in MOOCs do form community structures and that those structures are correlated with their learning [7, 4, 1, 11]. Brown et al. [1], for example, found that students connected to peers with a similar course performance.

While prior research has shown that students form stable social structures in MOOCs, the impact of those connections on students' performance has not always been consistent. Jiang et al. analyzed an algebra MOOC, and found that attributes of the students' social network were correlated with the students' final grades, but they found no relationship between the same variables in a different MOOC on finance [4]. Houston et al. likewise examined the forum activities that were most strongly associated with final grades in three different MOOCs and found that the addition of social centrality and similar features had no impact on their predictive models [3]. This inconsistency may be explained by the fact that the types of discussions students have and their relevance can change from class to class and that prior analyses have focused primarily on the overall social structure and not the content of the discussions. Though the peer communication features in MOOCs are intended to foster content engagement, many of the most active discussion-topics are often social conversations, critiques of the class videos, or exchanges of career advice [8]. Prior researchers have developed automated detectors to classify these posts into on- and off-topic comments and to evaluate the relative proportion of relevant discussions to learning outcomes [5, 10].

Our goal in this study is to examine how the topical content of forum discussions affects students' social relationships, as well as how they connect to their learning outcomes by addressing the following question: *Does the type (on- or off-topic) of conversation affect the relevance of students' social networks to their learning outcomes?*

2. DATA

For our analysis, we used data from a MOOC on "Big Data in Education" provided by The Teachers College at Columbia University and hosted on the Coursera (BDE 2013) and EdX (BDE 2015) platforms in 2013 and 2015 respectively. This

Yiqiao Xu, Niki Gitinabard, Collin Lynch and Tiffany Barnes
"What You Say is Relevant to How You Make Friends:
Measuring the Effect of Content on Social Connection" In:
*Proceedings of The 12th International Conference on
Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe
Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp.
679 - 682

was offered as an 8 week course that includes material from a graduate-level course on educational data mining and the analysis of big data in education. This curriculum introduces students to basic data collection and data analysis methods such as visualization and clustering. The course offers weekly lectures in the form of videos and individual assignments or quizzes which contribute to students' final grade (grade scale from 0 to 1). The students learn how and when to do educational data mining and learning analytics on data. The course was structured around weekly lecture videos and individual quizzes. The students' final course grade is a composite score based upon these quizzes. In the 2013 class, 1380 students completed at least one quiz, 778 students made at least one post or comment on the discussion forum producing a total of 603 discussion threads consisting of 4259 posts in total. In 2015, 320 students completed at least one quiz, 519 students produced 625 discussion threads with a total of 2056 posts. We manually annotated all of the posts and comments separating them into on- and off-topic entries after removing non-English posts from the dataset. Table 1 includes summary information about the number of students who received 0 or non-0 grades and posted on- or off-topic posts from both offerings of the course. Table 2 shows the number of posts and threads (defined by the starter post) of each type in each course.

	Content	0 grade	non-0 grade	Total
BDE 2013	on-topic	156	377	533
	off-topic	187	220	407
BDE 2015	on-topic	58	83	141
	off-topic	51	44	95

Table 1: Demographic of students grade and content

	Content	Post	Thread
BDE 2013	on-topic	2845	405
	off-topic	1388	380
BDE 2015	on-topic	1050	151
	off-topic	1006	367

Table 2: Number of on- and off-topic posts and threads

Tables 1 and 2 shows that in both courses, the students were more likely to take part in the on-topic discussions than the off-topic ones, especially for those who received non-0 grades. However, in 2015, we observed that though the on- & off-topic post counts are close to each other, the number of threads started with off-topic posts were smaller. This may be due to the fact that the EdX platform includes an optional private chat room for users, logs which we did not have access to.

3. METHODS

Prior researchers have shown that students' final grades are strongly related to those of their closest classmate or 'Best Friend' (BF) in traditional classroom [2]. This relationship also holds in online courses [11] and is also true for students' neighbors in a community structure [1]. In contrast to general assumptions the students are not always connecting with others who need help or people who share a goals or background but with people at their same level of perfor-

mance, in this study, we used exactly the same network generation approach as in our prior work [11] to build the social network graph and to evaluate the relationship between student communities and their final grades. And applied the Girvan-Newman algorithm with the "natural cluster number" approach described in [1] to identify coherent communities. We then applied the Kruskal-Wallis(KW) test to evaluate the correlation between clusters and performance.

In this approach, we treat forum participants as nodes, and we construct arcs between the individuals as weighted edges based upon their individual communications. In this approach we add a directed arc from the author's node to nodes representing the authors of all the comments that precede it in the thread. All of the forum contributors in the thread will be connected to one another. As the average thread length of our two datasets are 6.8 and 3.1 respectively, we developed this approach based upon the assumption that participants read the whole thread before they post any comments. Thus, we consider each reply to be an indication of an implicit social connection between forum participants. Once the raw directed graph has been constructed we modify the graph by eliminating all isolated nodes and merging the parallel edges to get a weighted undirected graph.

4. RESULTS

Table 3 shows the order (number of nodes) and size (number of edges) of the graphs that we obtained for the different content types and student groups (with/without 0 grade students) at the end of week-2. As we have established in prior work, the second week is considered the most important cut-off point for students to stay in or drop out of a course and to form their social networks [11]. We can observe that, in both courses, including the 0 grade students, students preferred to participate more in off-topic conversation than on-topic. According to the number of nodes and edges by the end of week-2, for an on-topic network, the edges still increase in frequency after week 2; however, for the off-topic posts, the social network has already formed at this time point. This suggests that the off-topic discussions may have been confined to a stable set of threads that only grew longer, or were confined to the same stable set of chatty people. Furthermore, for the non-0 students, they were more likely to start conversations for on-topic content, than the 0 students. One potential explanation for this is that students who did not plan to obtain a certificate, and who registered for free, participated in conversations such as introducing themselves to each other at the beginning of the course and then lost interest in the course. Another interpretation is that some of the students worked in spurts at the beginning but dropped out because the course did not fit their schedule over time. Our ongoing analysis of the forum content has shown that a number of the posts are also about early issues, such as course logistics and software. These issues became less relevant as the course progressed. Irrespective of the cause, the social structures are well established for off-topic discussions early enough that instructors should be able to provide advice early enough to the students who have lost interest early on.

Table 4 shows the number of students and the average final grade for each group of students with/without 0 grade. We found that students who participated in both the on-

	Content	Node*	Edge*	Grade	Node	Edge
2013	on-topic	392	2185	with 0	508	3778
				non-0	367	2713
	off-topic	356	9483	with 0	429	10389
				non-0	234	1884
2015	on-topic	182	637	with 0	199	721
				non-0	111	332
	off-topic	370	1044	with 0	392	1112
				non-0	98	95

Node*: Number of nodes at the end of week 2

Edge*: Number of edges at the end of week 2

Table 3: Graph order and size

and off-topic discussions received the highest average grade in the course. Students that only participated in off-topic discussions received the lowest final grade when compared to others. These results indicate that not only is sharing knowledge about course content is important, but participating in non-course content also has an impact on their learning process.

We also examined the growth rates of the number of posts, users and new threads as time progressed. We defined a new thread as being on- or off-topic based upon the head post that initiated it. For BDE 2013, on-topic posts and new threads increased over the whole course period, while off-topic posts appeared primarily at the beginning of the course before declining sharply. As for new users, they came in at the beginning to talk primarily on off-topic content, but new users were more likely to make on-topic conversation afterwards. However, for BDE2015, we observed that all three elements in the on-/off-topic social networks grew monotonically at a similar rate and that the number of new off-topic threads and users was always more than the on-topic ones. One of the potential reasons could be that students discussed course content topics in the private chat-room, rather than in the public forum. As one example, at the end of the BDE2013 course, there were 55,179 registered users, yet the final course social interaction graph contained only 778 participants, including 1 instructor and 2 teaching assistants. Some of the forum participants did not complete any quizzes, or even attempt to obtain the certificate, but still chose to engage in on- and off-topic discussions with others. On the other hand, some of the students who worked hard on the course did not contribute to the forum at all. There were 1,381 students who received a non-0 final grade; 934 of which did not post in the forum at all, while 304 zero final grade students did. It is conceivable students only posted when they faced particular difficulties, or that they sought help elsewhere as participation in the course forum was not a necessary condition for completion.

4.1 Social Interaction Analysis

As part of our analysis we also replicated the Best-Friends comparison as used by Brown et al. Here we identified each student's closest neighbor in the course, excluding members of the teaching staff, and then calculated a direct correlation between their grades and those of their best friends. Since the data was non-normal, we used Spearman's Rank Correlation Coefficient as a non-parametric test for association [9]. Our results are shown in Table 5.

	Content	Grade	BF	Comm
BDE 2013	on-topic	with 0	0.96	<0.05
		non-0	<0.05	<0.05
	off-topic	with 0	0.98	<0.05
		non-0	<0.05	0.08
BDE 2015	on-topic	with 0	0.24	<0.05
		non-0	<0.05	<0.05
	off-topic	with 0	0.06	<0.05
		non-0	0.38	0.30

Comm: KW test for community - grade

Table 5: Social connection correlation with content

Table 5 shows that the relationships between students' grades and those of their best friends were consistent between the traditional courses studied by Fire et al. [2] and MOOCs, but not immediately. Our results show that MOOC students, except those who did not submit any assignments, performed similarly to their closest peers.

5. CONCLUSION & DISCUSSION

In this paper, we distinguished posts and comments into on-topic (course relevant) and off-topic (non-course relevant) before analyzing students' social activities and their final grades. Interestingly, we drew a different conclusion from our previous work. In the BDE2013 dataset, including 0 grade students, we found no correlation between the students' closest 'best friend' and their performance, while the clustered community structures were significant related to their to performance. When we break this down into on- and off-topic networks respectively we found that there was a significant correlation with the community structure and grade for on-topic posts. For the off-topic, by contrast, only a moderate relationship was observed. For the BDE2015 non-0 students, the off-topic connection was not relevant to their performance. This is also shown by the average cluster grades for each group. This supports our original argument that the off-topic discussions may be confusing the social network analyses.

Additionally, Students who showed up in both the on- and off-topic discussions received the highest grade, higher than those that focused on the on-topic discussions alone. Students only made off-topic discussion received the lowest grades overall. Thus although participation in the on-topic discussion facilitated the students' learning, chatting with their peers on random topics was also relevant to their learning, albeit weakly. Additionally, according to Table 3, for all of the students, the off-topic graph was much bigger than the on-topic graph, while for the non-0 grade students, they were more likely to post course content topics than random chat. Thus, we conclude that, for the non-0 grade students who focused their efforts on finishing quizzes, passing the course, and receiving a certificate, participation on the forum helped them improve their grade and keep them from dropping out. By contrast, when we consider all of the students, including those with a 0 grade, the behavior of their closest peers does not seem to have affected them consistently. One possible explanation for this may be that 97% of the students received a 0 grade.

			on-topic	off-topic	only in on-topic	only in off-topic	in both on/off-topic
2013	with 0	students	508	429	299	220	209
		grade	0.51	0.38	0.44	0.16	0.61
	non-0	students	367	234	195	62	17
		grade	0.70	0.69	0.67	0.58	0.74
2015	with 0	students	199	392	100	293	99
		grade	0.33	0.14	0.26	0.05	0.40
	non-0	students	111	98	49	36	62
		grade	0.59	0.54	0.53	0.38	0.64

Table 4: Average grades for students in different social interaction

Moreover, as prior studies have shown the students formed the bulk of the social structures by the end of week 2. We found that 91% of the the off-topic connections had been formed by then. For the on-topic social network by contrast, only 57% of the connections had been formed by week 2. This highlights one of the limitations of prior work that conflated these social structures and it highlights the crucial importance of distinguishing posts by content. We observed different results for the BDE2015 dataset. This class had less proportion of on-topic discussions than BDE2013. This may be due in part to the fact that the edX platform provides support for private a chatrooms which students and instructors may use for side discussions. We were unable to access that data and it may be the case that much of the relevant communications were carried on there.

Discussion forums are widely used in MOOCs to support knowledge co-construction, but the connections between on-line social interaction and learning outcomes is still subject of some debate. As our study has shown the social network structures can be used for information provided we focus on the important on-topic discussions. And, as we have also shown it is possible to use trained models to support this classification, even ones that are trained in part across course offerings. Thus these findings highlight the critical importance of analyzing forum post content when exploring the relationship with learning outcomes; and to draw conclusions carefully when we work with datasets of this type.

In future work we plan to evaluate the impact of automatic classifiers and guidance, both for students and instructors, on students’ course performance, topic comprehension, dropout, and their final grades. This kind of help, we argue, will help students and instructors to manage the course content more effectively and will thus increase student engagement, reduce dropout, and improve other student outcomes.

6. ACKNOWLEDGEMENTS

This research was supported by NSF #1821475 “Concert: Coordinating Educational Interactions for Student Engagement” Collin F. Lynch, Tiffany Barnes, and Sarah Heckman (Co-PIs).

References

- [1] R. Brown, C. Lynch, Y. Wang, M. Eagle, J. Albert, T. Barnes, R. Baker, Y. Bergner, and D. S. McNamara. Communities of performance & communities of preference. In *EDM (Workshops)*, 2015.
- [2] M. Fire, G. Katz, Y. Elovici, B. Shapira, and L. Rokach. Predicting student exam’s scores by analyzing social network data. In *International Conference on Active Media Technology*, pages 584–595. Springer, 2012.
- [3] S. L. Houston II, K. Brady, G. Narasimham, and D. Fisher. Pass the idea please: The relationship between network position, direct engagement, and course performance in moocs. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, pages 295–298. ACM, 2017.
- [4] S. Jiang, S. M. Fitzhugh, and M. Warschauer. Social positioning and performance in moocs. In *Workshop on Graph-Based Educational Data Mining*, volume 14, 2014.
- [5] F.-R. Lin, L.-S. Hsieh, and F.-T. Chuang. Discovering genres of online discussion threads via text mining. *Computers & Education*, 52(2):481–495, 2009.
- [6] K. Reusser and C. Pauli. Co-constructivism in educational theory and practice. *International encyclopedia of the social & behavioral sciences*, 3:913–917, 2015.
- [7] C. P. Rosé, R. Carlson, D. Yang, M. Wen, L. Resnick, P. Goldman, and J. Sherer. Social factors that contribute to attrition in moocs. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 197–198. ACM, 2014.
- [8] D. T. Seaton, Y. Bergner, I. Chuang, P. Mitros, and D. E. Pritchard. Who does what in a massive open online course? *Commun. ACM*, 57(4):58–65, Apr. 2014.
- [9] P. Sedgwick. Spearman’s rank correlation coefficient. *BMJ: British Medical Journal (Online)*, 349, 2014.
- [10] A. F. Wise, Y. Cui, and J. Vytasek. Bringing order to chaos in mooc discussion forums with content-related thread identification. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*, pages 188–197. ACM, 2016.
- [11] Y. Xu, C. F. Lynch, and T. Barnes. How many friends can you make in a week?: evolving social relationships in moocs over time.
- [12] D. Yang, T. Sinha, D. Adamson, and C. P. Rosé. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Proceedings of the 2013 NIPS Data-driven education workshop*, volume 11, page 14, 2013.

Deep-IRT: Make Deep Learning Based Knowledge Tracing Explainable Using Item Response Theory

Chun-Kit Yeung
Find Solution Ai Limited
Hong Kong
chunkit@findsolutiongroup.com

ABSTRACT

Deep learning based knowledge tracing model has been shown to outperform traditional knowledge tracing model without the need for human-engineered features, yet its parameters and representations have long been criticized for not being explainable. In this paper, we propose *Deep-IRT* which is a synthesis of the item response theory (IRT) model and a knowledge tracing model that is based on the deep neural network architecture called dynamic key-value memory network (DKVMN) to make deep learning based knowledge tracing explainable. Specifically, we use the DKVMN model to process the student's learning trajectory and estimate the item difficulty level and the student ability over time. Then, we use the IRT model to estimate the probability that a student will answer an item correctly using the estimated student ability and the item difficulty. Experiments show that the Deep-IRT model retains the performance of the DKVMN model, while it provides a direct psychological interpretation of both students and items.

Keywords

Knowledge tracing, item response theory, deep learning.

1. INTRODUCTION

Generally, the knowledge tracing task can be formalized as follows: given a sequence of student's historical interactions $\mathbf{X}_t = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t)$ up to time t on a particular learning task, it predicts some aspects of his next interaction \mathbf{x}_{t+1} . Question-and-answer interactions are the most common type in knowledge tracing, and thus \mathbf{x}_t is usually represented as an ordered pair (q_t, a_t) which constitutes a tag for the question q_t being answered at time t and an answer label a_t indicating whether the question has been answered correctly. In many cases, knowledge tracing seeks to predict the probability that a student will answer a question q_{t+1} correctly given the sequence \mathbf{X}_t , i.e., $P(a_{t+1} = 1 | q_{t+1}, \mathbf{X}_t)$.

Many mathematical and computational models have been

developed to solve the knowledge tracing task. These models can be grouped into two categories [3]: (1) a highly structured model whose parameters have a direct meaningful interpretation, e.g., Bayesian knowledge tracing (BKT) [2] and performance factors analysis (PFA) [4]; (2) a highly complex but general-purpose model whose parameters are difficult to interpret, e.g., deep knowledge tracing (DKT) [5] and dynamic key-value memory network (DKVMN) for knowledge tracing [9]. The former category typically provides more insight besides the prediction result, while the latter usually performs better without requiring substantial feature engineering by humans. To the best of our knowledge, there has not yet been a model that is highly complex and general-purpose, yet simultaneously explainable. Therefore, it is appealing to devise a model that inherits the merits of these two categories.

In this paper, we propose *deep item response theory* (Deep-IRT) to make the deep learning based knowledge tracing model explainable. The Deep-IRT model is inspired by the Bayesian deep learning [7] and is a synthesis of a deep learning model and a psychometric model. Specifically, the Deep-IRT model utilizes the DKVMN model [9] to process input data and return psychologically meaningful parameters of the IRT model [6]. The DKVMN model performs feature engineering job to extract latent features from student's historical question-and-answer interactions. Then, the extracted latent features are used to infer the difficulty level of and the student ability on each KC over time. Based on the estimated student ability and the KC difficulty level, the IRT model predicts the probability that the student will answer a KC correctly. By formulating the knowledge tracing task with both the DKVMN model and the IRT model, we are getting the merits from these two models. The Deep-IRT model benefits from the advance of deep learning techniques, e.g., capturing features that are hard to be human-engineered. On the other hand, we empower the explainability by introducing a well-known psychometric model which can be easily understood by many people.

2. DEEP ITEM RESPONSE THEORY

The Deep-IRT architecture at time t is visualized in Figure 1.¹ Firstly, the knowledge state of each latent concept

¹The detail working mechanism of DKVMN can be found in [9]. In summary, at time t , the DKVMN model first receives a KC q_t , then predicts the probability of answering q_t correctly, and eventually updates the memory using the question-and-answer interaction (q_t, a_t) .

Table 2: The average test results of the evaluation measures, as well as their standard deviations, from 5 trials are reported. As the PFA model can be learned by a closed-form solution, the learned parameters and thus its performance are the same in every trail. Therefore, the standard deviation is not reported for the PFA model.

Dataset	PFA			DKVMN			Deep-IRT		
	AUC	Acc	Loss	AUC	Acc	Loss	AUC	Acc	Loss
ASSIST2009	59.68	69.24	7.08	81.61 \pm 0.06	77.01 \pm 0.04	5.29 \pm 0.01	81.65 \pm 0.02	77.00 \pm 0.06	5.30 \pm 0.01
ASSIST2015	52.85	73.37	6.13	72.94 \pm 0.06	75.18 \pm 0.03	5.71 \pm 0.01	72.88 \pm 0.07	75.14 \pm 0.02	5.72 \pm 0.01
Statics2011	64.99	79.85	4.64	83.17 \pm 0.11	81.57 \pm 0.05	4.24 \pm 0.01	83.09 \pm 0.12	81.56 \pm 0.04	4.24 \pm 0.01
Synthetic	61.68	65.20	8.01	82.97 \pm 0.06	75.58 \pm 0.07	5.62 \pm 0.02	82.98 \pm 0.07	75.61 \pm 0.04	5.61 \pm 0.01
FSAL-F1toF3	54.52	54.57	10.46	68.40 \pm 0.89	63.40 \pm 0.15	8.42 \pm 0.03	68.69 \pm 0.28	63.43 \pm 0.24	8.42 \pm 0.06

learning tablet application called 4LittleTrees².

3.2 Results

The model performance of the experiment is shown in Table 2. In addition, we also include PFA model [4] in Table 2 as a baseline model for reference.

All in all, the DKVMN and Deep-IRT models have a similar performance. We conduct two-tailed independent t-tests on each dataset and each evaluation measure between the Deep-IRT model and the DKVMN model. We found that the difference between their performance is not significant for majority of the datasets (with p-value > 0.1). Although we cannot claim that their performance is, more or less, the same based on the large p-value, this result, however, might imply that the Deep-IRT model potentially retains the performance of the DKVMN model.

4. GOING DEEPER IN DIFFICULTY LEVEL

To evaluate the KC difficulty estimated from the Deep-IRT model, we compare the difficulty level learned for the FSAL-F1toF3 dataset with four other sources. The reason why we use the proprietary dataset is that we have the individual questions' difficulty level provided by the publisher. Each question is associated with a difficulty level in $\{1, 2, 3\}$ which represents easy, medium and hard, respectively.

The second source of difficulty level is calculated according to the item analysis [1]. The difficulty level of the item analysis is the percentage of students who answer a question correctly in a test environment. Yet, to be consistent with interpretation with other models, we adopt the percentage of students who answer a question *incorrectly*. Furthermore, as our dataset is not collected from a test environment, a student can answer a same question multiple times. Thus, we only adopt the student's first attempt when calculating the difficulty level. Moreover, we only consider the question on which at least 10 distinct students has answered.

The third source of difficulty level is learned by the one-parameter IRT model. We also use the student's first attempt, only, on a question to learn the IRT model for the sake of avoiding multiple attempts on the same question. Moreover, we adopt $\sigma(\theta_i - \beta_j)$ to be the item response function when learning the IRT model.

Lastly, we trained a PFA model to extract the difficulty level

²More information about the 4LittleTrees can be found on <https://www.4littletrees.com/>.

in a knowledge tracing setting, rather than a test environment setting. In other words, we use the entire student's learning trajectory to learn the question's difficulty level.

Since there are more than two thousands questions in this dataset, we only evaluate a set of questions that belong to a subset of skills. This subset contains five skills that constitute around a fifth of the interactions in the FSAL-F1toF3 dataset, and has in total 131 questions. These skills are "Significant Figures", "Approximation and Errors in Measurement", "Index Notation", "Laws of Indices" and "Polynomials". We visualize the difficulty level obtained from different sources in a pairs plot in Figure 2 with the Pearson correlation r stated in the lower triangular part of the pairs plot. The positions in the pairs plot are ordered according to the evaluation setting and the complexity of getting the difficulty level.

The pairs plot reveals that the difficulty level learned from the Deep-IRT model aligns with most of the other sources with a strong correlation, except for the difficulty level provided by the publisher. Moreover, it is observed that the more similar the models' evaluation setting and complexity are, the higher the Pearson correlation is between the models. For example, the Pearson correlation between the difficulty level from the item analysis model and the IRT model is 0.96, while the Pearson correlation between the item analysis model and the Deep-IRT model is 0.56. Furthermore, it is observed that the difficulty level provided by the publisher is moderately correlated to the one obtained from the item analysis model (0.40) and the IRT model (0.39), but weakly correlated to the one obtained from the Deep-IRT model (0.08). Thus, it would be interesting to examine whether the difficulty level inferred from the Deep-IRT model would be more accurate than other traditional models.

5. CONCLUSION

In this paper, we propose the Deep-IRT model which empowers the deep learning based knowledge tracing model with explainability. Experiments show that the Deep-IRT model retains the performance of the deep learning based knowledge tracing model while simultaneously being able to estimate the KC difficulty level and the student ability level over time. Moreover, the difficulty level estimated by the Deep-IRT model aligns with the difficulty level obtained by other traditional methods, e.g., the IRT model and the item analysis. Thus, it potentially provides an alternative way to estimate KC's difficulty level by utilizing the entire learning trajectory, rather than the traditional educational testing environment.

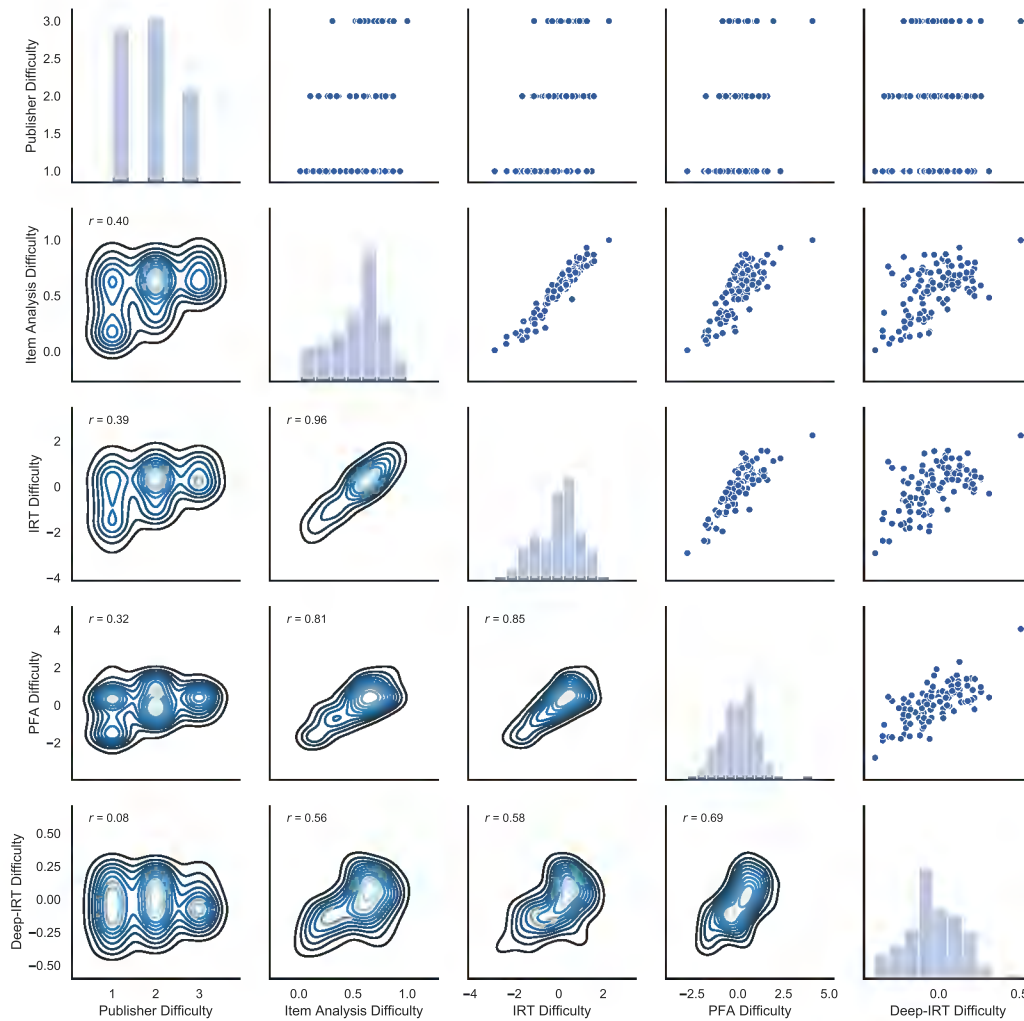


Figure 2: Pairwise comparison of the difficulty level obtained from different sources. The positions in the pairs plot are ordered according to the evaluation setting and the complexity of getting the difficulty level.

References

- [1] Understanding Item Analyses. Office of Educational Assessment, University of Washington. Retrieved February 11, 2019 from <http://www.washington.edu/assessment/scanning-scoring/scoring/reports/item-analysis/>.
- [2] Albert T. Corbett and John R. Anderson. Knowledge tracing: modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4 (4):253–278, March 1995.
- [3] Mohammad M. Khajah, Robert V. Lindsey, and Michael C. Mozer. How deep is knowledge tracing. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 94–101, 2016.
- [4] Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. Performance factors analysis – a new alternative to knowledge tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, pages 531–538, Amsterdam, Netherlands, 2009. IOS Press.
- [5] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J. Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.
- [6] Georg Rash. Probabilistic models for some intelligence and attainment tests. *Copenhagen: Danish Institute for Educational Research*, 1960.
- [7] Hao Wang and Dit-Yan Yeung. Towards Bayesian deep learning: A framework and some existing methods. *IEEE Transactions on Knowledge and Data Engineering*, 28 (12):3395–3408, 2016.
- [8] Frances M. Yang and Solon T. Kao. Item response theory for measurement validity. *Shanghai Archives of Psychiatry*, 26(3):171–177, 2014.
- [9] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 765–774. International World Wide Web Conferences Steering Committee, 2017.

Accurate Modelling of Language Learning Tasks and Students Using Representations of Grammatical Proficiency

Ahmed H. Zaidi
Computer Laboratory
University of Cambridge
ahz22@cl.cam.ac.uk

Russell Moore
Computer Laboratory
University of Cambridge
rjm49@cam.ac.uk

Andrew Caines
Computer Laboratory
University of Cambridge
apc38@cam.ac.uk

Paula Buttery
Computer Laboratory
University of Cambridge
pjb48@cam.ac.uk

Christopher Davis
Computer Laboratory
University of Cambridge
ccd38@cam.ac.uk

Andrew Rice
Computer Laboratory
University of Cambridge
acr31@cam.ac.uk

ABSTRACT

Adaptive learning systems aim to learn the relationship between curriculum content and students in order to optimise a student's learning process. One form of such a system is content recommendation in which the system attempts to predict the most suitable content to next present to the student. In order to develop such a system, we must learn reliable representations of the curriculum content and the student. We consider this in the context of foreign language learning and present a novel neural network architecture to learn such representations. We also show that by incorporating grammatical error distributions as a feature in our neural architecture, we can substantially improve the quality of our representations. Different types of grammatical errors are automatically detected in essays submitted by students to an online learning platform. We evaluate our model and representations by predicting student scores and grammatical error distributions on unseen language tasks.

1. INTRODUCTION

In general the adaptive learning approach has been shown to lead to improved learning outcomes for student users of educational platforms [4, 8, 10]. However, there remains a question of what is the best methodology to construct representations for students and tasks. Previous approaches manually engineer features to construct representations [7]. These features are usually tuples of a knowledge component (e.g. differentiation, fractions in the case of maths) and student outcome (i.e. whether or not the student demonstrated understanding for that knowledge component through completing the task). A task may contain multiple knowledge components. Whilst this approach is highly interpretable, in the domain of language learning, it is difficult to clearly

divide the tasks into knowledge components. Furthermore, in the recently popular paradigm of deep learning, we have seen that training representations through neural networks have yielded state-of-the-art results in the space of image recognition, and various natural language tasks.

Motivated by this, we propose a methodology of automatically developing high quality representations of students and tasks in a language learning context. Having reliable student and task representations in place facilitates work on downstream tasks such as curriculum learning and recommender systems for language learning.

Representations are derived from a novel neural architecture and real student data collected through the Write & Improve¹ (W&I) assessment and feedback platform for learners of the English language. [13].

Our best-performing model incorporates grammatical error distributions detected by ERRANT [3] as a feature and achieves mean squared error (MSE) of 1.195 on score prediction, an absolute value of 1.093 on a scoring scale of 0-13.

2. WRITE & IMPROVE

On W&I, students are prompted to input a short text of at least 25 words in response to a given question. Once they have completed the task, the W&I automarker assigns each text an integer score s between 0 and 13. The system also automatically provides a grade on the CEFR scale² along with feedback on grammatical errors detected in the text. Table 1 outlines how integer scores are mapped to the CEFR scale. Currently all users of W&I move through the curricu-

Ahmed Zaidi, Andrew Caines, Christopher Davis, Russell Moore, Paula Buttery and Andrew Rice "Accurate Modelling of Language Learning Tasks and Students Using Representations of Grammatical Proficiency" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 687 - 690

Table 1: Student scores mapped to CEFR levels

A1	A2	B1	B2	C1	C2
1-2	3-4	5-6	7-8	9-10	11-13

¹<https://writeandimprove.com>

²The Common European Framework of Reference for Languages

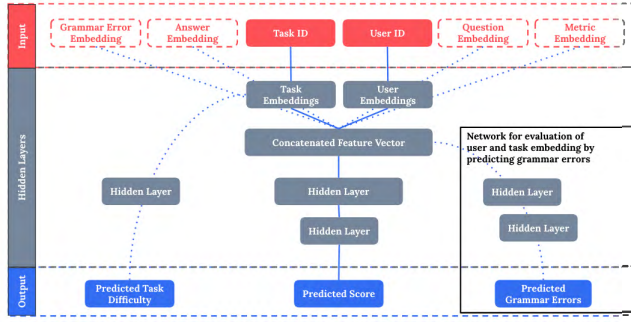


Figure 1: Task score prediction system architecture. Dotted lines and boxes are optional features and network connections.

lum in an unguided and independent fashion. An intelligent tutoring system would instead guide students from task to task in order to personalise their learning experience and improve their level of performance. In order to do that we must learn reliable student and task representations.

We obtained application logs of user activity from the past two years – a total of 3+ million essay submissions by 300k account holders. We filtered the data for users who had submitted at least 10 submissions. We also had a record of the questions (‘prompts’) users responded to and the scores assigned to their texts by W&I’s auto-marker.

3. LEARNING STUDENT AND TASK REPRESENTATIONS

Our primary goal was to predict student scores on a given language learning task based on our representations of students and tasks in W&I. Secondary to that, we check the quality of our student representations by predicting the grammar error distribution of a given student-task tuple. In what follows we describe the data, evaluation metrics and models used in this work.

3.1 Model Architecture

The architecture of our neural system can be seen in Figure 1. The neural network takes as an input a user id u and task id t which are taken as indices in the user embedding layer U and task embedding layer T respectively. $u \in N_u$ where N_u is the number of unique users in the W&I dataset. $t \in N_t$ where N_t is the number of unique tasks in the W&I dataset.

We optimise our system and learn a user embedding matrix U and task embedding matrix T by minimising the mean squared error (MSE) of our predicted score s and the target score \hat{s} .

We introduce an auxiliary objective to predict the difficulty β of each task t , referenced as t_β . The ground-truth labels for task difficulty (beginner, intermediate, advanced) are obtained from the meta-data of each task in the W&I dataset.

3.2 Feature Set

In addition to the score s , the W&I dataset contains prompts and answers in natural language as well as metrics on whether

submission k is the highest scoring submission by user u . We incorporate these additional features into the architecture of the model in order to evaluate their impact on the quality of user and task embeddings.

3.2.1 Answer and Question Embedding

We obtain a vectorised form of each student response and question using 300-dimension word2vec embeddings³ pre-trained on the Google News corpus [5]. Our embeddings are an additive compositional model where the final embedding is a sum of every word in the question or answer. Whilst this model is not state-of-the-art for distributional semantics, Mitchell & Lapata [6] show that the additive model can yield results comparable to more sophisticated models.

3.2.2 Metric embedding

The metric embedding is a 2-dimensional vector. The first dimension is a binary value for whether the score for the submission was the highest score on task t for user u . The second dimension is a binary value for whether the score for the submission was the highest score across all W&I tasks for user u .

3.2.3 Grammar error embedding

A student’s grammatical proficiency plays a vital role in determining how well they perform on a particular task. As we do not know of any system that identifies appropriate use of grammar, we focused on understanding what grammatical structures the student struggles with. This was done by running ERRANT [3], an automated error detection and correction system, in order to identify grammatical errors in the student’s essay.

For each submission k , we constructed a 47-dimensional vector, one dimension for each of the error types observed in the W&I dataset. Each dimension stored the number of times that error type appeared in the student’s essay submission.

$$\langle e_k \rangle = \langle f_k^1, f_k^2, \dots, f_k^{47} \rangle \quad (1)$$

– where e_k is the grammar error embedding e for submission k , and f_k^n is the frequency of errors for error type n in submission k .

3.3 Mean score baseline

Our baseline system for predicting s for user u on task t is to calculate the mean of observed scores by all users for that task. We refer to this baseline as MEAN_SCORE.

3.4 Evaluation

We identify two approaches to evaluating our system and the quality of our learned user and task representations: 1) score prediction; and 2) grammar error prediction.

³A word2vec embedding is a $1 \times x$ dimensional dense vector that represents a word semantically.

Table 2: Score prediction (MSE) and grammar embedding prediction (cosine) results for the top 8 best performing feature combinations (error: grammar error embedding; ques: question embedding; ans: answer embedding; metric: metric embedding).

Model	MSE	Cosine
MEAN_SCORE (baseline)	1.913	-
error+ques+ans+metric	2.254	-0.385
ques+metric	1.942	-0.402
ans+metric	1.951	-0.414
error+metric	1.350	-0.426
ques	2.028	-0.403
ans	2.014	-0.412
error	1.761	-0.410
metric	1.907	-0.393

3.4.1 Evaluation of score predictions

To evaluate the performance of score prediction we use mean squared error (MSE) in common with other works in this field, using global computation where all data points are treated equally [9].

3.4.2 Evaluation of grammar embedding predictions

In order to further evaluate the quality of the learned user and task representations, we also introduce an additional evaluation task of predicting the distribution of grammar errors for a user u on a task t .

This was done by building a network that takes as an input the user \vec{u} and task \vec{t} from the pre-trained embedding U and T and predicts the grammar embedding \vec{g} . Our dataset for grammar error prediction was created by extracting the last submission k of every user u . This was to ensure that the system is predicting the distribution of errors for the users at their most recent knowledge state.

We optimise our system by minimising the cosine proximity of the predicted grammar vector \vec{g} and the target grammar vector $\hat{\vec{g}}$.

4. RESULTS

Table 2 summarises the results of our system. We compare the effectiveness of various features in the prediction of a user's score s on a task t which is evaluated by MSE. We include the top 8 MSE values on the score prediction system and their corresponding cosine value from the grammar error prediction model.

We find that incorporating question and answer embeddings do not provide any performance improvement in terms of MSE beyond the baseline model. The metric embedding provides marginally better results than the baseline with an MSE of 1.907. The grammatical error embedding provides substantial improvements beyond both the baseline and the metric embedding with an error of 1.761. The best performing system incorporates both grammatical error embedding and metric embedding, reducing the MSE to 1.350.

Table 3: Performance across various student and task representations sizes (N_h)

Model	N_h	MSE	Cosine
error+metric	3	1.350	-0.426
error+metric	5	1.297	-0.431
error+metric	16	1.245	-0.415
error+metric	32	1.195	-0.433

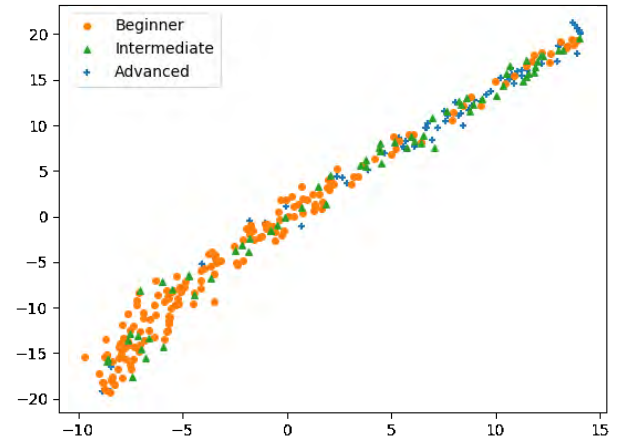


Figure 2: t-SNE of 300 randomly sampled student representations classified by different levels of proficiency

Table 3 shows the model that provides the lowest cosine proximity to the target grammatical error vector (i.e. best system) was error+metric, which is consistent with the lowest MSE for the score prediction system.

In order to interpret the relevance of cosine proximity we conducted a Pearson's correlation test between the MSE values from the score prediction system and the cosine proximity scores from the grammar error prediction system. The results show a 0.7883 Pearson's correlation with a p -value of 0.0201 which is statistically significant at $\alpha < 0.05$.

Figure 2 shows a t-SNE [12] of 300 randomly sampled student representations learned by our best performing score prediction system. The students are classified by their proficiency which has been determined by observing the most frequent task level attempted in their five most recent submissions. Qualitatively, the results from the plot are promising as the advanced and intermediate users, whilst present throughout the plot, are more concentrated towards the top right (higher level of language proficiency). Beginner students, on the other hand, are more concentrated in the bottom left. This suggests that the embeddings constructed from our model provide context on the language abilities of the student.

5. DISCUSSION

The results in Table 2 show that incorporating grammar error embeddings provides a reliable signal to learn well-

Student Knowledge Diagnosis on Response Data via the Model of Sparse Factor Learning

Yupei Zhang, Huan Dai, Yue Yun, and Xuequn Shang^{*}
Northwestern Polytechnical University
1 Dongxiang Road
Xi'an, China
{ypzhaang,shang}@nwpu.edu.cn

ABSTRACT

Cognitive diagnosis aims to analyze the status of knowledge mastery of student and is thus very important for personalized education. The existing methods mostly depend on the empirical Q-matrix from domain experts. However, the knowledge points in Q-matrix are unavoidably overlapping, leading to the weak performance on the practical applications. In this paper, we propose a novel model for student knowledge diagnosis, called Sparse Factor Learning (SFL). SFL learns a meta-knowledge dictionary from student response data of test questions, where the knowledge structure of any entity (e.g., student, question or others) is a sparse linear combination of dictionary atoms. Our method has three innovations for cognitive diagnosis: learning latent nonoverlapping meta-knowledge, sparsely representing the entities, and removing the bias noise for guessing and slipping. To verify our method, we collected the response data from the final exam of C language program of international class and then conducted the experiments for knowledge diagnosis, student grouping and response prediction. The experiment results show that SFL works effectively and results in decent performance. Besides, it delivers that student who favors mathematics and physics can achieve higher score. All codes and data set can be available on our website.¹

Keywords

Personalized education, Sparse factor learning, Cognitive diagnosis Student modeling, Student grouping

1. INTRODUCTION

In recent years, educational data mining (EDM) is an emerging research field which seeks to develop methods for exploring response data of learners. Researchers in this field aim to discover the true knowledge structure of learners to offer learners accurate assistance. Cognitive diagnosis, a hotspot

issue in EDM research, explains the mental processes that are triggered when test items are solved [2, 5].

Recent progress has been made on applying machine learning algorithms to mine true and latent knowledge structure of learners. These existing cognitive diagnosis methods propose a probability model reflecting learner response data. In contrast to rule-based approaches, machine learning-based models promise to be rapid and inexpensive to deploy. However, latest research proposes “abstract concepts” [1] which are unavoidably overlapping, leading to the intricate results. In addition, it cannot analyze unseen data. Therefore, accurate teaching problem still remains an open issue.

In this paper, motivated by recent progress, we present SFL, a new model to best distinguish learner using their nonoverlapping knowledge structure. This model defined as knowledge learning is designed as sparse representation for student response, which is analogous to dictionary learning [6]. We first observe the data $y_{ij} = 1$ or 0 depending on whether learner j answers question i correctly or incorrectly. SFL model learns the latent nonoverlapping meta-knowledge spaces of certain course and can also represent the real knowledge structure of learners by optimizing a novel objective function. Furthermore, the sparse representation of learners in the meta-knowledge dictionary can be obtained. After that, we cluster learners who have the same meta-knowledges according to this sparse representation and predict whether a learner can response the question correctly. More specifically, our contributions are listed as follows:

- We tackle the weakness of traditional educational models which focus on Q-matrix given by experts. However, the acquisition of the Q-matrix is not easy anytime. Our model still discover essential knowledge structure without Q-matrix.
- We propose a data-driven method called SFL to explore the correlation between inseparable meta-knowledge and learner from student response data. SFL removes the bias noise for guessing and slipping. This model provides natural explanatory power in learning process.
- We apply SFL model to real-world teaching work. We collect more complete data from Northwestern Polytechnical University and draw more meaningful educational conclusion from data.

^{*}Corresponding author

¹<http://www.nwpu-bioinformatics.com/>.

The rest of paper is organized as follows. In Section 2, we formally put forward to our targeted issue and mathematical model. Section 3 details the algorithm of our SFL model. Simulation and real data experimental results are presented in Section 4. Finally, conclusions are given in Section 5.

2. METHODOLOGY

In this section, we formalize our problem in detail, and propose new method termed Sparse Factor Learning.

2.1 Model for Learner Response Data

In this section, we set forth the definition of meta-knowledge. Meta-knowledge is virtual and *inseparable* atom that learner cannot master. It is believed that all questions and learner knowledge structure consist of these meta-knowledges.

In this problem, we have N learners that answer Q questions. As shown in Figure 1, we are provided with the data \mathbf{Y} from the response of learners over a collection of questions, where $y_{ij} = 1$ or 0 indicating correct or not of the learner j at the question i , respectively. In Figure 2, the matrix \mathbf{D} denotes the relationship between the collection of questions and meta-knowledges where the coefficient d_{ij} indicates the associate degree of the i -th question over the j -th meta-knowledge point. By the way, we denote the correlation between meta-knowledges and learners by matrix \mathbf{X} . Usually, the data is often polluted by subjective response. We therefore consider this educational case by assuming that the i -th learner has guessing and slipping rate on the j -th question by w_{ij} . Thus, we represent observed data over the dictionary in terms of $\mathbf{D}\mathbf{X} + \mathbf{W}$. Our problem could be defined as follow:

Problem Definition 1: Given the learner response data, i.e. binary matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n] \in R^{Q \times N}$, the primary purpose of our sparse factor learning framework is : (1) learning the overcomplete basis $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_k] \in R^{Q \times K}$, where we assign $K \gg Q$ to obtain association between knowledge latent attributes and questions; 2) obtaining the sparse representation matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n] \in R^{K \times N}$ which reflects the connection between meta-knowledges and learners; (3) estimating the bias matrix \mathbf{W} which is a sparse bias matrix.

2.2 The Proposed model

Since real data is binary data, SFL model can be formulation as the following equation shows:

$$\mathbf{Y} = \text{sign}(\mathbf{D}\mathbf{X} + \mathbf{W}) \quad (1)$$

where $y_{ij} \in \{0, 1\}$ is the result of learner j at question i . Utilizing the model (1) and the given observations of question-learner responses \mathbf{Y} , our goal is to estimate the factors $\mathbf{D}, \mathbf{X}, \mathbf{W}$. To solve problem, we exploit these observations as prior information: (1) Meta-knowledges are generally large and overcomplete. (2) Questions are linear combination with meta-knowledges. (3) Each column of matrix \mathbf{X} is sparse and matrix \mathbf{W} is sparse.

3. ALGORITHM

Under the prior information, we propose a new method dubbed Sparse Factor Learning (SFL) and solve this problem using ADMM framework[3].

	Q questions				
N learners	1	0	1	1
	1	0	1	1
				
	1	0	1	1

Figure 1: question-learner response.

	Meta-knowledges				
Q questions	0.032	0.045	0.012	0.003
	0.126	0.007	0.028	0.001
			d_{ij}	
	0.003	0.021	0.056	0.006

Figure 2: question i involves meta-knowledge j

3.1 Problem Formulation

To estimate \mathbf{D}, \mathbf{X} , and \mathbf{W} , we minimize the objective function as follow:

$$\begin{aligned} \min_{\mathbf{D}, \mathbf{X}, \mathbf{W}} \sum_{i,j} \left\| y_{ij} - \text{sign}(\mathbf{d}_i^T \mathbf{x}_j + w_{ij}) \right\|_F^2 \\ \text{subject to } \|\mathbf{d}_i\|_2 \leq 1, \|\mathbf{x}_j\|_0 \leq s, \|\mathbf{W}\|_0 \leq k \end{aligned} \quad (2)$$

Complied with prior information (3), we impose sparsity on each vector \mathbf{x}_j . We constrain $\|\mathbf{x}_j\|_0 \leq s$ to limit maximum number of nonzero coefficients. Unfortunately, these constraints lead to an NP problem. Hence, (2) cannot be solved efficiently in practice. In order to arrive at an optimization problem that can be solved with a reasonable computational complexity, we relax the sparsity constraint $\|\mathbf{x}_j\|_0 \leq s$ in (2) to L_1 -norm constraint as in [1]. Because the function $\text{sign}(x)$ is not a continuous derivable function, we then use $\text{sigmoid}(x)$.

$$\text{sign}(x) \approx \text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (3)$$

By introducing Lagrange multipliers, our objective function can be reduced into:

$$\min_{\mathbf{D}, \mathbf{X}, \mathbf{W}} \left\| \mathbf{Y} - \text{sigmoid}(\mathbf{D}\mathbf{X} + \mathbf{W}) \right\|_F^2 + \lambda \sum_{j=1}^M \|\mathbf{x}_j\|_1 + \beta \|\mathbf{W}\|_1 \quad (4)$$

where regularization terms $\sum_{j=1}^M \|\mathbf{x}_j\|_1$ and $\|\mathbf{W}\|_1$ induce sparsity on each vector \mathbf{x}_j and matrix \mathbf{W} , and parameters $\lambda > 0$ and $\beta > 0$ are to control the sparsity level.

3.2 The Sparse Factor Learning Algorithm

Inspired by most of dictionary learning algorithm which fix one variable updating another variable, SFL adopts an alternating optimization approach named alternating direction method of multipliers (ADMM) to solve (4). We introduce an auxiliary variable \mathbf{J} in order to make the objection

function of (4) separable. The optimization problem can be rewritten as follows:

$$\begin{aligned} \min_{\mathbf{D}, \mathbf{X}, \mathbf{W}} \quad & \|\mathbf{Y} - \text{sigmoid}(\mathbf{DX} + \mathbf{W})\|_F^2 + \lambda \sum_{j=1}^M \|\mathbf{x}_j\|_1 + \beta \|\mathbf{W}\|_1 \\ \text{s.t.} \quad & \mathbf{W} = \mathbf{J} \end{aligned} \quad (5)$$

The augmented Lagrangian function of problem (5) is

$$\begin{aligned} \mathcal{L}(\mathbf{D}, \mathbf{X}, \mathbf{W}, \mathbf{J}, \mathbf{M}) = & \|\mathbf{Y} - \text{sigmoid}(\mathbf{DX} + \mathbf{W})\|_F^2 + \lambda \sum_{j=1}^M \|\mathbf{x}_j\|_1 \\ & + \beta \|\mathbf{J}\|_1 + \langle \mathbf{M}, \mathbf{W} - \mathbf{J} \rangle + \frac{\mu}{2} \|\mathbf{W} - \mathbf{J}\|_F^2 \end{aligned} \quad (6)$$

where M is Lagrange multiplier and $\mu > 0$ is a penalty parameter. The variables are updated alternately by minimizing its corresponding problem with fixing other variables. The iteration stops when the convergence conditions are arrived. The detailed procedure of solving the proposed SFL problem is described in Algorithm 1.

Algorithm 1 Sparse Factor Learning

Input: Data Matrix \mathbf{Y}

Parameter: $\mathbf{D} = \mathbf{0}, \mathbf{X} = \mathbf{D}^{-1}\mathbf{Y}, \mathbf{W} = \mathbf{0}, \lambda = 0.03, \beta = 0.4$

Output: $\mathbf{D}_{k+1}, \mathbf{X}_{k+1}, \mathbf{W}_{k+1}$

- 1: while not converged ($k = 1, 0, \dots$) do
- 2: fix the others and update \mathbf{J}

$$\mathbf{J}_{k+1} = \max\{\mathbf{S}_{\frac{\beta}{\mu}}(\mathbf{W}_k + \frac{1}{\mu}\mathbf{M}_k)$$

fix the others and update \mathbf{W} :

$$\mathbf{W}_{k+1} = \mathbf{W}_k - \alpha \frac{\partial \mathcal{L}(\mathbf{W}_k)}{\partial \mathbf{W}_k}$$

- 3: fix the others and update \mathbf{X} [4]by:

$$\mathbf{x}_j = \text{BIHT}(\mathbf{y}_j, \mathbf{D}) = \mathbf{H}_M(\mathbf{x}_k + \mathbf{D}^T(\mathbf{y}_j - \text{sigmoid}(\mathbf{D}\mathbf{x}_k + \mathbf{W})))$$

- 4: fix the others and update \mathbf{D} by:

$$\mathbf{d}_{k+1} = \mathbf{d}_k - \beta \frac{\partial F(\mathbf{d}_k)}{\partial \mathbf{d}_k}$$

- 5: update the multipliers:

$$\mathbf{M}_{k+1} = \mathbf{M}_k + \mu_k(\mathbf{W}_{k+1} - \mathbf{J}_{k+1})$$

μ can be a constant value.

- 6: check convergence

$$\frac{\|\mathbf{Y} - \text{sigmoid}(\mathbf{DX} + \mathbf{W})\|_F^2}{\|\mathbf{Y}\|_F^2} < \varepsilon$$

- 7: End while.
-

4. EXPERIMENTS

In this section, we validate SFL on real-world educational data sets. We conduct the two practical experiments: student grouping and response prediction. Meanwhile, we compare SFL with SPARFA method which is a pioneer work. All experiment codes are implemented by MATLAB R2018a.

NPU-C Dataset: Student data used to test our model is collected from Northwestern Polytechnical University across students in international class who entered the C Language Program final exam in December 2018. First, we collected the response of students of 20 objective questions and the answers to these questions are only right or wrong. The dataset contains two elements: 0 for incorrect answer and 1 for correct answer which is collected by thirty-nine students at twenty questions. Second, to collect the background information, we made a questionnaire with twenty questions containing gender, age, English and Math grade of College Entrance Examination, their favorite subjects, and so on. Here, there is no missing data in the response matrix. More details can be found in our website.

4.1 Student Grouping on Our Data

After obtaining the sparse representation of all students, we utilize k-means and spectral clustering to group the student who has same knowledge structure. We use Jaccard distance as metric and measure our results employing Dun Validity Index (DVI) and Compactness Index (CP), defined as follow:

$$\text{DVI} = \frac{\min_{0 < m \neq n < k} \left\{ \min_{\forall x_i \in \Omega_m, \forall x_j \in \Omega_n} \|x_i - x_j\| \right\}}{\max_{0 < m \leq k} \left\{ \max_{\forall x_i, \forall x_j \in \Omega_m} \|x_i - x_j\| \right\}} \quad (7)$$

The numerator computes the minimum distance between each pair of clusters, while the denominator calculates the maximum distance between all pair of two elements in a cluster. That is, the bigger DVI value corresponds the better cluster.

$$\text{CP} = \frac{1}{k} \sum_{i=1}^k \left(\frac{1}{|\Omega_i|} \sum_{x_i \in \Omega_i} \|x_i - w_i\| \right) \quad (8)$$

where k means the number of cluster and w_i represents the central point corresponding to a certain cluster. The term in bracket is to compute the distance between each point and its central point. Thus, the smaller CP value delivers the better cluster.

We vary the different number of clusters $\{2, 3, 4, 5\}$ and record our results in Table 1 and 2. From Table 1 and 2, we can conclude that our model obtains better clustering performance than using original data directly and the SPARFA. Besides, we choose to use k-means and spectral clustering methods to gather the response data into three clusters. We use DVI evaluation to measure the results and list our results in Table 3 and 4. Table 3 and 4 illustrate that using original data or SPARFA method is not good comparing to SFL method. In addition to these cluster indicators, we draw conclusions from our data (see more in our website): (1) The first group indicates that students who is good at Mathematics and more like Mathematics and Physics will achieve better grades than others in general. (2) The second group shows that students of being weak on Mathematics and Physics seem to be weak on C Language Programming. (3) The third cluster contains many complex reasons depending on their learning. We observed that exercise and study hard after class have a great effect on the final grades.

Table 1: Cluster results using DVI metrics

	2	3	4	5
Original	0.427	0.302	0.333	0.316
SPARFA	0.119	0.119	0.132	0.135
SFL	0.545	0.536	0.535	0.536

Table 2: Cluster results using CP metrics

	2	3	4	5
Original	1.9271	2.0438	1.9905	2.2603
SPARFA	1.1932	1.9995	2.5085	2.8598
SFL	0.83842	0.82933	0.80897	0.80996

Table 3: Comparisons using different protect cluster method with CP.

	Original	SPARFA	SFL
K-means	2.044	2.000	0.829
Spectral	0.180	0.1653	0.162

Table 4: Comparisons using different cluster method with DVI.

	Original	SPARFA	SFL
K-means	0.462	0.382	1.991
Spectral	0.302	0.119	0.494

4.2 Student Response Prediction

For predicting the responses of a student at questions, we here determine the responses on two factors: the knowledge structure of student composed of few meta-knowledge points and the meta-knowledge involved in questions.

In this experiment, we vary different data sets of different size from $\{5, 10, 15, 25, 35, 39\}$. For each data set, we hold out one of the learner responses as testing set, and train SFL on the rest to acquire the question-metaknowledge matrix \mathbf{D} and the metaknowledge-learner matrix \mathbf{X}_{train} to obtain the above two factors. Because the size of our data is small, we hence use leave-one-out (LOO) validation to achieve more effective evaluations. To obtain more stable result, we run 20 times to achieve the average values shown in Figure 3. In addition, we use the neighborhood classifier for prediction on original data and the SPARFA method to predict student response. We observe that the performance of all of algorithms generally improves as the data size increases. Moreover, the error rate of using original data is not reduced much, and Figure 3 suggests that the error rate of SPARFA method is not reduced much when dataset is small. Generally, the average result of our method performs better. This indicates that our model is more effective.

5. CONCLUSION

In this paper, we remedy the shortcomings of traditional cognitive diagnosis model by proposing a data-based method, called Sparse Factor Learning (SFL) model. SFL has three innovations. First, we exploit the natural concept named meta-knowledge and test paper, questions and knowledge points

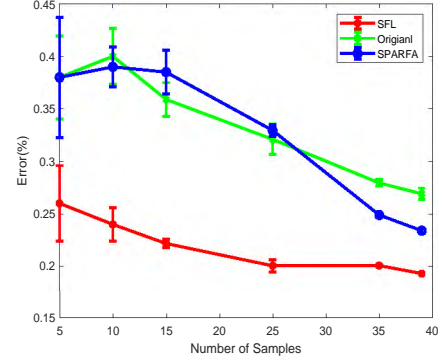


Figure 3: Student Response Prediction.

from field experts can be thought as a linear combination with the meta-knowledges. In addition, using the meta-knowledges dictionary, we can have a clear representation of student response data. Second, it is believed that learner can master limited meta-knowledges from one course, and hence we have a sparsity constraint on learner-knowledge matrix. Third, we remove the bias noise of guessing and slipping. Experiments on our practical data show that SFL is effective and has good generalization ability for prediction. In future, we will test our method using more data and design a reasonable strategy for missing data.

6. ACKNOWLEDGMENTS

The research has been supported by The National Natural Science Foundation of China (No.61802313, No.U1811262) and The Fundamental Research Funds for the Central Universities (No.G2018KY0301).

7. REFERENCES

- [1] Andrew S. Lan, Andrew E. Waters, Christoph Studer, and Richard G. Baraniuk. Sparse factor analysis for learning and content analytics. *Journal of Machine Learning Research*, 15(1):1959–2008, 2013.
- [2] Antonio Preti, Marcello Vellante, and Donatella R. Petretto. The psychometric properties of the aReading the mind in the eyes test: an item response theory (irt) analysis. *Cognitive Neuropsychiatry*, 22(3):233–253, 2017.
- [3] Yu Wang, Wotao Yin, and Jinshan Zeng. Global convergence of admm in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, 78(1):29–63, 2019.
- [4] Hadi Zayyani, Mehdi Korki, and Farrokh Marvasti. Dictionary learning for blind one bit compressed sensing. *IEEE Signal Processing Letters*, 23(2):187–191, 2015.
- [5] Peida Zhan, Hong Jiao, Manqian Liao, and Yufang Bian. Bayesian dina modeling incorporating within-item characteristic dependency. *Applied psychological measurement*, 43(2):143–158, 2019.
- [6] Yupei Zhang, Ming Xiang, and Bo Yang. Graph regularized nonnegative sparse coding using incoherent dictionary for approximate nearest neighbor search. *Pattern Recognition*, 70:75–88, 2017.

Collaboration Analysis Using Object Detection

Zhang Guo
Computer and Information Sciences
University of Delaware
guozhang@udel.edu

Roghayeh Barmaki
Computer and Information Sciences
University of Delaware
rlb@udel.edu

ABSTRACT

The analysis of the collaborative learning process is one of the growing fields of education research. In this paper, we present a new method to automatically evaluate the collaboration level of students in groups using computer vision techniques. Instead of single-time solicitation techniques such as self-reported questionnaires to evaluate the collaboration, we evaluate the entire process of the collaboration by analyzing image data captured from groups. We use object detection techniques such as Mask R-CNN method to process our data. This process is based on detecting people and other objects from pictures and video clips of the collaborative learning process, then evaluating the performance of students in the interactive learning and textbook settings using these collaborative indicators. Two collaborative indicators are related to the team's proximity and time on task. We tested our approach to evaluate the group-work collaboration in a controlled study of 33 teams while performing an anatomy muscle painting intervention. The results indicate that our approach successfully recognizes the differences of collaborations among teams of treatment and control conditions, $F_{(1,33)} = 11.42, p < 0.005$. This work may offer implications for automated quality prediction of collaborations among human-human interactions in group-work scenarios.

Keywords

Collaborative learning, collaborative learning analysis, Mask R-CNN, object detection

1. INTRODUCTION

Collaborative learning is a widely-used education pattern featured by small group interaction and team-based evaluation metrics. Typically two or more participants are assigned in the same group and work for a common purpose, which encourages them to learn via teamwork [6]. Researches in the past few years have shown that compared with individual learning or lecture-based learning, collaborative learning as an active learning approach can increase stu-

dents' learning motivation and improve knowledge retention. However, it is undeniable that these benefits only work on the well-formed teams with respectful members which have efficient collaborative activities during the learning process.

In early attempts to analysis collaboration, researchers either established effective collaborative learning models or built reasonable standards for judging the collaborative learning process based on self-reported survey data or collaboration system data. However, those data were collected from class attendance, quizzes scores, and reports which could not directly reflect participants' interaction during the learning activities. The ideal data should reflect the students' actions and emotions during the process, which could be recorded via images or videos. Also considering the learning-based methods, tremendous data can make the model general. On the other hand, there is no objective measurement to automatically evaluate the process of collaboration while students performing group work. One possible evaluation method is defining new criteria using the features extracted from the image or video records.

For image and video data analysis of group-work quality identification, we introduce a new object detection approach to extract useful features such as identifying participants, and objects in the scene. In this study, our goal is using object detection to detect participants and their locations from the image data and then recognize collaborative actions from these proximity features. We use a pre-trained Mask R-CNN (Region-based Convolutional Neural Network, [5]) with the COCO dataset [7] as our object detection approach. We evaluated the usefulness of the Mask R-CNN object detection method in identifying collaborations in a tangible, muscle painting activity.

2. CASE STUDY

We conducted a between-subjects study while performing anatomy learning intervention (muscle painting) in a large laboratory course of General Biology in spring 2018 at Johns Hopkins University. Students worked in pre-assigned teams to complete the muscle painting activity. During the muscle painting activity, pairs of students collaborate to paint twelve muscles of their body using painting supplies. The first student plays the role of a model, while her teammate, as a painter, locates the major upper-limb muscles using the human anatomy diagram in the lab manual [8], and then paints her upper limb. Afterwards, students switch roles, and the upper limb painter becomes a model for the lower

Zhang Guo and Roghayeh Barmaki "Collaboration Analysis Using Object Detection" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 695 - 698

limb. The painting activity was upgraded for the treatment group by using mobile tablet devices instead of the textbook for visualizing the musculoskeletal system. Figure 1 shows two settings of the case study during the muscle painting activity.

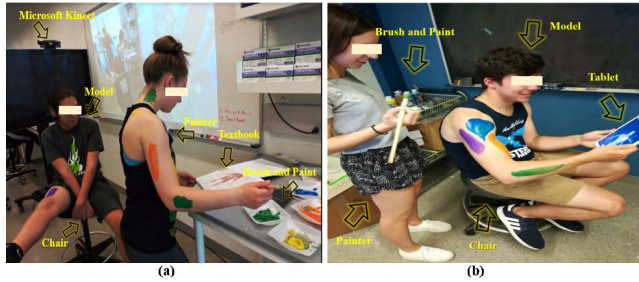


Figure 1: Case study setup for participants in pairs to complete the painting activity using either (a) textbook, or (b) tablet.

There was a workstation for each laboratory room to capture videos and photos so that students' team performance can be recorded during the painting activities.

The study was approved by the Institutional Review Board (Protocol HIRB00005021), and oral informed consent was obtained from each participant before the case study commenced. After consent, students entered the painting activity room with their teammates and completed the painting task. All students completed pre and post questionnaires before and after the intervention.

3. DATASETS

Image data. Image data is the pictures and video clips captured from the muscle painting scene every ten-seconds (Figure 3(a) shows one of the snapshots from the scene). Only group members and the tools they used for painting activity were captured in the image data.

Approximately 2000 images was captured in total from 33 teams. Based on different time spent on the painting activities, data for each team had about 25 to 200 images. Each image file was also timestamped for further time on task analysis.

COCO dataset. COCO dataset is a large online open-source dataset and contains photos of 80 easily recognized common objects categories with instance-level segmentation mask, including person, bottle, book, and cellphone [7].

Survey data. Survey data was collected online, generated by the Qualtrics application, towards all participants before and after completing the painting activities. There were two questionnaires in this study. Pre-questionnaire consisted of demographic questions and a pre-test. Post-questionnaire was composed of questions about participants' user experience during collaboration study, including the preference of being a painter or a model, the level of engagement in the activity, and a post-test. This data set will be used in our future work.

4. ANALYSIS OF COLLABORATION

4.1 Object Detection

Object detection is a computer technology related to computer vision and image processing that deals with identifying the instance of semantic objects of a certain class in digital images and videos [10]. It can do both localization and classification. One object detection strategy is to localize each object with a bounding box and do classification on each bounding box area [9]. Thus, object detection methods can help instructors detect participants and extract useful features for analysis from the raw images.

Mask R-CNN approach. Mask R-CNN approach uses the Region Proposal Network (RPN) stage to generate bounding box for each candidate object and uses a more accurate module RoIAlign to extract features from each box and show the bounding-box regression and classification [5] (see Figure 2). By feeding the input image, a CNN feature extractor is able to extract image features which are called feature maps. Then a CNN RPN will create Region of Interests (RoI) which are the candidate object regions generated by RPN and ranked based on their score (how likely is the candidate object region could contain an object). Then the $N = 300$ regions with the highest scores are kept. Each of them will be warped into fixed dimension by RoIAlign and feed into three parallel branches: two Fully Connected (FC) layers as Faster R-CNN make classification and boundary box prediction and two additional convolution layers to build the binary masks. The top 100 detection boxes are kept and form a $100 \times 80 \times 28 \times 28$ tensor, where 80 represents the number of classes in the COCO training dataset, and 28×28 represents the size of each predicted mask. The resized masks and the bounding boxes can be overlaid on the original input image as a transparent layer.

Using a pre-trained neural network by COCO dataset [1], we are able to apply Mask R-CNN to automatically detect participants, and their locations from the collaborative learning activities image data as shown in Figure 2. In our implementation, we focused on the following features of the object detection results (Figure 3(b)): image file name, category name, bounding box coordinates and the confidence score for each object. These features are the key points for collaborative learning analyses in our muscle painting scenario. We detected the participants with person confidence score ≥ 0.9 .

4.2 Collaboration Metrics

To evaluate students' performance during our collaborative learning process, we defined two indicators: Level of collaboration, and Time on Task. Level of collaboration is formulated by Overlapping area and ratio.

Level of Collaboration. In muscle painting activity, collaboration was crucial in terms of how closely the participants work together. Especially close physical distance or proximity, and the amount of time students work together to perform the task was part of the learning process. We measured this factor using Mask R-CNN approach and named it level of collaboration. Once again, we want to highlight that our assumption may only fit for tangible work-group activities like muscle painting with pairs of students, and may not

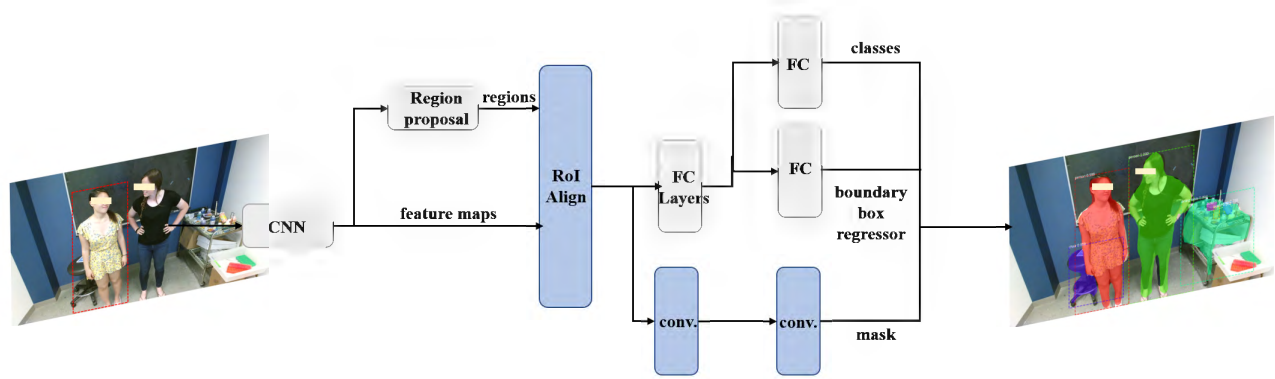


Figure 2: The Mask R-CNN framework for object detection.

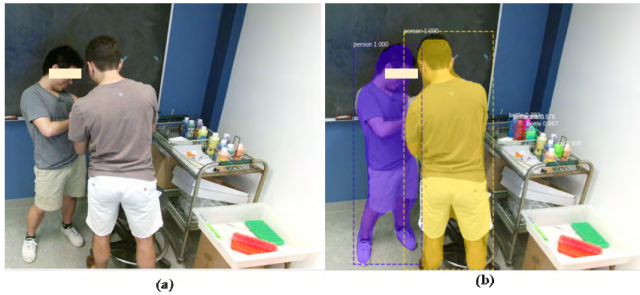


Figure 3: Collaborative learning using tablet in painting activity (a) original image data, and (b) object detection using Mask R-CNN.

be generalized to other types of group-work activities which need distributed task allocations.

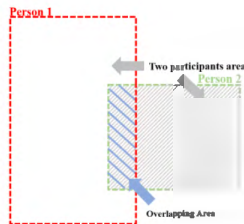


Figure 4: The overlapping ratio is equal to the overlapping area divided by the smaller participant area.

Overlapping Area and Ratio for Level of Collaboration. The idea for computing overlapping area and ratio came from how close two participants were during the painting activity. Since there are lots of body contacts between painters and models while collaboration study, their bounding boxes may overlap. Using the coordinates of the bounding boxes of the person instance, we calculate the overlapping area (if their bounding boxes do not overlap at all, the overlapping area and ratio will be zero) and the area for each participant (see Figure 4). The overlapping ratio is equal to the overlapping area divided by the smaller participant area. The reason for comparing the overlapping area with the smaller participant area is to eliminate the bias that

one participant may have a larger box than the other. For each group, compute the mean of all the overlapping ratio in the object detection results as the level of collaboration value.

Time on Task. We also calculated the time that students dedicated to performing the collaborative painting task. Previous work [2] shows that increased time on task is a strong predictor of knowledge retention. Un-engaged students just finish the activity in the minimum possible time. In this case, we consider time on task as one of the indicators of student engagement as well. The longer time they spend, the better the collaborative performance they have. This indicator is computed by the file names of the image data and the number of images for each group (aka time-stamped file information).

Since the treatment group use the mobile learning technology [4] during the muscle painting activity, we hypothesize that teams of students who use the tablet have better collaborative performance than the control group. In comparison to the control group who use the textbook, students in the treatment group collaborate more with each other within proximate distance and they may spend more time on task in the collaborative activity.

5. RESULTS

Participants. Total of 66 (39 Female) students in 33 teams participated in this study. 17 out of 33 pairs were in the treatment group (14 Female), and 16 pairs were in the control group (25 Female). A summary of the results from these participants is reported in Table 1.

Level of Collaboration. Among 33 teams, we calculated level of collaboration using the aforementioned method, and compared the treatment group versus the control group. The results showed that there was a statistically significant difference between these two groups based on level of collaboration; $F_{(1,33)} = 11.42$, $p < 0.005$, Cohen's $d = 1.18$ (large effect size).

Time on Task. Similarly, treatment group had a meaningful difference with control group based on time on task $F_{(1,33)} = 4.29$, $p < 0.05$, Cohen's $d = 0.72$ (relatively large effect size).

Table 1: Summary of the descriptive analytic for two groups of the study.

Groups	Level of Collaboration (%) ¹	Time on Task (seconds)
	Mean ($\pm SD$)	Mean ($\pm SD$)
Treatment	9.23 \pm 2.85	631.18 \pm 353.75
Control	6.39 \pm 1.84	428.75 \pm 170.64

¹Level of collaboration was measured based on the percentage of overlapping area of students while working on the painting task.

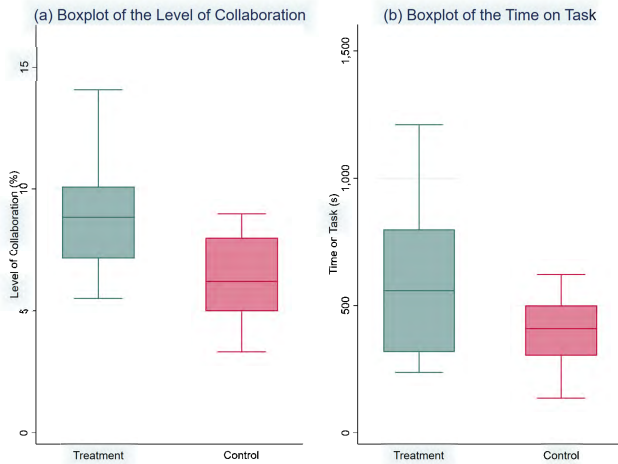


Figure 5: The box plots of Level on Collaboration and Time on Task for treatment and control groups in the study.

According to the results, collaboration level and time on task had large effect size values and high level of significance which indicated that both two indicators have identified the variations among treatment and control conditions successfully. Based on Figure 5 (a and b), the treatment group had higher means and more able to achieve higher value in both indicators. That means, by using mobile tablets, students were able to get closer to each other and would like to spend more time to complete the task than use textbooks. During the activity, teams with the tablet were more likely to share the contents and discuss with teammates in a close distance. Therefore, we could recognize the treatment performed better than the control groups during collaborative learning. Thus, our pre-trained Mask R-CNN approach can provide useful features, and can be used for collaborative learning analysis with acceptable accuracy.

6. CONCLUSION AND FUTURE WORK

In this paper, we used Mask R-CNN, an object detection approach, to analyze the quality of collaboration. We then evaluated the approach with two collaborative indicators related to team's proximity and time on task. The results showed that our approach was capable of recognizing differences in the level of collaboration among students in treatment versus control groups. Both the time on task and level of collaboration could successfully distinguish the differences between two groups of the study.

The focus of current project is understanding collaboration on muscle painting activity. Some aspects of our work could

be improved in the future. We plan to use our survey data as another dimension to perform collaboration evaluation. We also aim to hand-annotate part of our collected data based on the object categories we need, including painting brush, and re-train our model on the entire data set. We will work on other collaborative features, such as facial expression recognition, emotion recognition, head, and body pose estimation, and joint attention estimation to better understand the level of collaborations among teams. Finally, we aim to use a 3D pose estimation method of the human body using a single image [3] in our future work to better understand complex interpersonal collaborations with computer vision techniques.

7. ACKNOWLEDGEMENTS

We thank Drs. Rebecca Pearlman, and Richard Shingles for integrating our study to their General Biology Lab course, laboratory technicians, and teaching assistants for assisting us on data collection, and study participants for their contributions to making this study possible. We also wish to acknowledge the support from our research collaborators Dr. Nassir Navab, and Kevin Yu for their insight in study design and development.

8. REFERENCES

- [1] W. Abdulla. Mask R-CNN for object detection and instance segmentation on keras and tensorflow, 2017.
- [2] R. Barmaki, K. Yu, R. Pearlman, R. Shingles, F. Bork, G. M. Osgood, and N. Navab. Enhancement of anatomical education using augmented reality: An empirical study of body painting. *Anatomical sciences education*, 2019.
- [3] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it smpl: Automatic estimation of 3D human pose and shape from a single image. In *European Conference on Computer Vision*, pages 561–578. Springer, 2016.
- [4] W. S. Cheung and K. F. Hew. A review of research methodologies used in studies on mobile handheld devices in k-12 and higher education settings. *Australasian Journal of Educational Technology*, 25(2):153–183, 2009.
- [5] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
- [6] D. W. Johnson and R. T. Johnson. Cooperation and the use of technology. *Handbook of research for educational communications and technology: A project of the Association for Educational Communications and Technology*, pages 1017–1044, 1996.
- [7] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.
- [8] E. N. Marieb and P. B. Jackson. *Essentials of Human Anatomy and Physiology Laboratory Manual*. Pearson/Benjamin Cummings, 2006.
- [9] M. Riesenhuber and T. Poggio. Models of object recognition. *Nature neuroscience*, 3(11s):1199, 2000.
- [10] M. Sonka, V. Hlavac, and R. Boyle. *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

Design of an Elective Course Recommendation System for University Environment

Boxuan MA
Graduate School of Information Science and Electrical Engineering
Kyushu University
Fukuoka, Japan
ma.boxuan.611@s.kyushu-u.ac.jp

ABSTRACT

Course recommendation system is a useful tool that not only helps the students who have no sufficient experience to decide what they should study, but also leverages their full performance if they could study what they like or are interested in. Different from MOOCs, the selection and recommendation for hybrid learning environments such as university are relatively difficult. Students who enrolled in the same course may have completely different purposes and different interest. Employing the enrollment record data from Kyushu University, we conduct a systematic investigation on the course-taking pattern for recommendation. We then discuss the challenges to recommend suitable courses in university and propose a preliminary approach to address the challenges by designing a course recommendation mechanism based on association rule of previous course-taking pattern together with student interest.

Keywords

Learning Analytics; Hybrid Learning Environments; Data Mining; Association Analysis; Recommender Systems

1. INTRODUCTION

Course recommendation is considered a challenging domain that could help students in suggesting suitable courses for them as well as reducing time to explore courses that they will take. To provide suggestions to assist learners who have no sufficient experience to choose courses which they need, researchers have tried to develop various approaches to recommend relevant learning materials to learners for MOOCs. Although MOOCs have their own benefits, they are inherently limited in supporting learners in physically co-located environments such as classrooms and group/peer learning spaces. The selection and recommendation at university are often different from those in MOOCs.

Courses in university environments are closely interwoven with various types of physical, pedagogical and social contexts. There are required courses and elective courses in university. Students generally receive a grade and academic credit after completion of the course and they need to get enough credits to meet the requirements of graduation. It is difficult for students to decide which courses they should take because there are a

large number of elective courses opened in each semester, they have to spend a lot of time for exploring those courses, and they may not be able to explore all of them. Also, student interest and goal can change as they explore and discover something meaningful on and off campus, and there are complex constraints and contexts that have to be considered in choosing courses.

(1) Complex constraints

Different from MOOCs, various courses are provided for each academic year, however, students usually do not choose many courses because learning a course is a time-consuming task. In Kyushu University, for instance, statistics show that each course usually lasts for several weeks and a student enrolls in 14 courses on average each year. Moreover, students may take courses in the first two years mostly, because they may busy for internship or finding jobs in the third and fourth year.

(2) Anti-interest

Students may not choose courses based purely on their interest in university environment. For example, some students would not enroll in a course which contains contents they are interested in, they just choose the course that allow them to get credits easily. In addition, student enrollment behavior may be influenced if they are not familiar with the contents of those courses.

(3) Long-tail effect

Long-tail effect is a classic problem in recommendation system. For Kyushu University, figure 1 shows the long-tail distribution of course popularity. There are several popular courses which will be filled up quickly while most of courses will not be selected by students frequently. Those courses may disappear because there are not enough students follow.

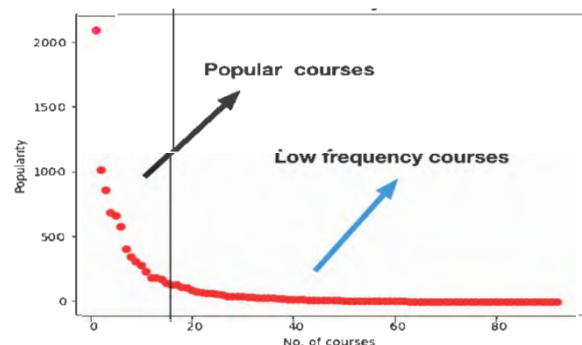


Figure 1. Long-tail distribution of course popularity.

Boxuan Ma "Design of an Elective Course Recommendation System for University Environment" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 699 - 701

2. RELATED WORK

Various approaches have been used in applications for course recommendation. Content-based filtering approach recommends an item to a user by considering the description of the item and clustering the item and the user into groups to gain similarity between them. Piao and Breslin [1] propose a strategy to extract user profile text from their LinkedIn pages and to calculate the similarity with course profile text, then give the recommendation results by the similarity. Apaza et al. [2] use LDA to train two different topic models on both college course syllabus and online course syllabus then a content-based match algorithm is used to estimate the ratings from a user to all courses.

Collaborative filtering approach recommends an item to a user by investigating the user's similarity with the user's information in a system and predict the item that the user would be interested in. Hana [3] introduces a mechanism based on this approach to recommend courses for a student by exploring the student's academic record and matching the record with others' ones to gain the similarity. Then the system figures out and recommends which course he is good at or interested in so that he could pass the course. Elham S.Khorasani et al. [4] proposes a Markov Chain Collaborative Filtering model to recommend courses based on historical academic data with concerns of the sequence of each course being taken. X.Jing and J.Tang [5] use LDA to extract user-specific latent information from their historical access behaviors to represent their interest profile. Then the similarity is calculated from above interest profile, and recommend outcomes are given based on the similarity. Kiratijuta et al. [6] developed an elective course recommendation system which investigates each student's academic records to find the similarity with the targeted student.

Association rules based on frequent patterns are often used to discover interesting relations between variables in databases. In terms of course enrollment, the objective is to extract rules from data that describe previous course selections from students in higher education. Aher and Lobo [7] use association rule mining together with clustering to recommend courses by using historical data. Narimel Bendakir and Esma A. Imeur [8] presented a course recommendation system, which incorporates association rule and user ratings in recommendation. Parameswaren et al. [9] propose a course recommendation mechanism for university students, which consider complex constraints.

Compared with previous studies, the main contribution of our work is that we exploit student interest and use that to improve the traditional association rule. In addition, we use other information including course relation and social contexts to boost the recommendation performance. We designed our framework to combine them in order to make better use of available information.

3. PRELIMINARYWORK

This is a research project in its earlier stages. Currently, we collected approximately 38,968 pseudonymized enrollment records from 2,366 students. These students enrolled in 2015 at Kyushu University. And then we propose a preliminary approach and try to address those challenges above. In particular, we first analyzed the records of university student course-taking patterns by using the association rule analysis. Besides, learner

interest is used to better reveal students' potential choice and increase the diversity and abundance of recommendation results.

3.1 Association rule

It is useful for taking advantage of the collaborative experience of the students who have finished their studies. However, in the traditional association rule method, the importance of each item in the database is the same, this kind of course recommendation based on support threshold and confidence threshold is not enough. For example, some courses with good contents are excluded because the amounts of selection are lower than the support threshold. Also, the popular courses have been frequently extracted since they have been selected many times, but students may not have much interest in them. As a result, most of the recommended courses come from the relatively popular courses, however, those courses are not necessarily the ones that students are most interested in. Finally, students may be misled to choose those popular courses and other courses may disappear because there are not enough students follow.

In our framework, we plan to use interest to reflect the different importance of each course to recommend suitable contents for students. The purpose is to recommend a wider range of courses to students, avoiding the problem of insufficient interest caused by traditional association rules.

3.2 Measures of Interest

Student interest is the most important target of our framework. We want to build a simple but universal representation of student interest which will be useful not only in course recommendation but also in other personalization tasks. Characterizing what is interesting is a difficult problem. The very definition of the word "interest" is elusive. A common solution in e-learning system is directly combining user enrollment and the tags of enrolled courses [5]. While this solution suffers from the small size of tag set and the sparsity of enrollment. In our framework, we extract student interest from student activities of courses which contain much more abundant information.

Measures of interest include objective measures and subjective measures. Objective measures of interest or objective interest criteria are measures of interest that depend only on the data and the patterns extracted from it, for example, the rating of a course from student. Subjective measures of interest or subjective interest criteria are measures that also depend on the specific needs and prior knowledge of the student.

Firstly, we divided students into different clusters based on their department. Each cluster has its own preference on courses enrolling according to the enrollment records. Then every student can take the average rating of students who belong to the same cluster as an interest weight on every course. Then, available information including student attendance, quizzes, learning activities during class and their final score in each course could be used to calculate the subjective measures. Moreover, both teachers and students write e-journals after each class, which are helpful to get feedback. This kind of feedback will imply the interest of students for a specific course, which could be considered as objective measures together with student goals.

Figure 2 illustrates an overview of the recommendation system. We first use association rule together with student interest to get rules for a certain cluster based on their department. The rules'

consequents determine a list of courses to be recommended. Then we plan to use social search and courses prerequisite to adjust the final results.

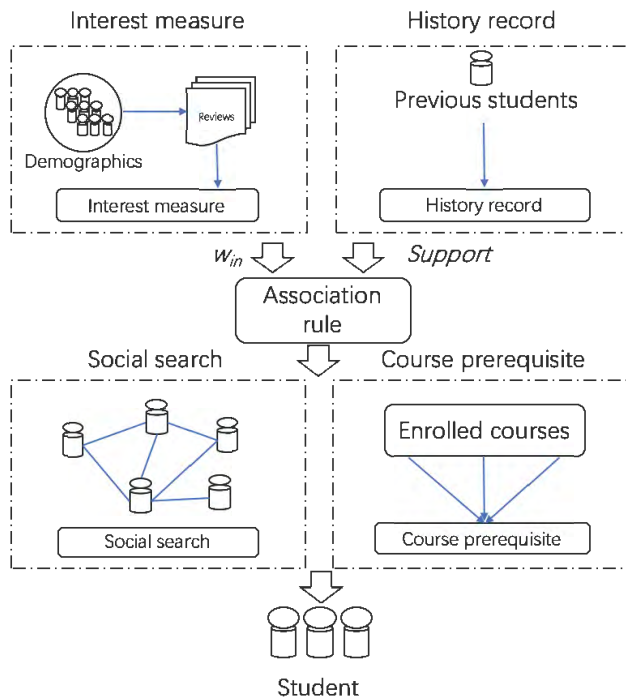


Figure 2. Overview of the recommendation framework.

4. FUTURE DIRECTION

This research aims to discuss the difference between MOOCs and university environment to recommend suitable courses for learners and study how to improve personalized course recommendation in such environment. Apart from assisting the student, it also will help the university registrar to manage courses. In particular, we propose a preliminary approach which extracts student-specific interest from e-learning system and combines association rule and student interest together to give recommend results. In addition, our approach would be helpful in other personalization situations.

As future work, social course search and course prerequisite will be used to boost our method for university environment, we will compare our framework with other recommendation system to get the evaluation result. In addition, mining other information from student behaviors to match the contents of learning materials is also an important work to do.

5. ADVICES SOUGHT

For this doctoral consortium, advice is sought regarding some concerns.

- Is this idea both significant and valuable? For example, could it be applied in a broader range of domains?*
- What kind of student data should be used for a course recommendation system? For now, we give the results only based on historical data from previous students, who have already finished the course.*
- Are there any other methods for predicting student interest for a certain course? Since the information from current student is limited, so what kind of data should be useful?*
- Are there any suggestions for the evaluation procedures of this research?*

6. REFERENCES

- Guangyuan Piao and John G Breslin. 2016. Analyzing MOOC Entries of Professionals on LinkedIn for User Modeling and Personalized MOOC Recommendations. In Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization. ACM, 291–292.
- Rel Guzman Apaza, Elizabeth Vera Cervantes, Laura Cruz Quispe, and José Ochoa Luna. 2014. Online Courses Recommendation based on LDA.. In SIMBig. 42–48.
- BYDŽOVSKÁ, Hana. Course Enrollment Recommender System. Proceedings of the 9th International Conference on Educational Data Mining. Raleigh, NC, USA: International Educational Data Mining Society, 2016. s. 312-317.
- Elham S.Khorasani, Zhao Zhenge, John Champaign. A Markov Chain Collaborative Filtering Model for Course Enrollment Recommendations: 2016 IEEE International Conference on Big Data (Big Data), P. 3484 – 3490.
- Jing,X.,Tang,J. :Guess you like: course recommendation in Moocs. In: Proceedings of the International Conference on Web Intelligence, ACM (2017) pp. 783–789.
- Bhumichitr, Kiratijuta et al. “Recommender Systems for university elective course recommendation.” 2017 14th International Joint Conference on Computer Science and Software Engineering (JCSSE) (2017): 1-5.
- Sunita B Aher and LMRJ Lobo. 2013. Combination of machine learning algorithms for recommendation of courses in E-Learning system based on historical data. Knowledge-Based Systems 51 (2013), 1–14.
- Bendakir, Narimel, and Esma Aïmeur. “Using association rules for course recommendation.” Proceedings of the AAAI Workshop on Educational Data Mining. Vol. 3. 2006.
- Aditya Parameswaran, Petros Venetis, and Hector Garcia-Molina, “Recommendation systems with complex constraints,” ACM Transactions on Information Systems, 29(4), 1–33, 2011. <http://doi.org/10.1145/2037661.2037665>.

Towards Modeling Students' Problem-solving Skills in Non-routine Mathematics Problems

Huy Nguyen
Carnegie Mellon University
hn1@andrew.cmu.edu

John Stamper
Carnegie Mellon University
jstamper@cs.cmu.edu

Bruce M. McLaren
Carnegie Mellon University
bmclaren@cs.cmu.edu

ABSTRACT

A key issue in mathematics learning and instruction is supporting high-school students in developing problem-solving skills, which can foster creativity, innovation, and ultimately lead to success in the workplace. Towards this goal, a crucial first step is modeling students' problem-solving skills, so that effective instructions and interventions can be designed to meet their learning needs. We propose three steps to explore this direction in the domain of non-routine mathematics problems -- those that cannot be solved by a textbook procedure but require insight and resourcefulness. First, we construct a conceptual framework of the most common problem-solving strategies that our instruction and modeling process will be based on. Second, we conduct a pilot study to test whether the framework can help students generate multiple solution strategies. Third, we use natural language processing and crowdsourcing to support automated assessment of students' inputs and explanations. These three steps will contribute to an educational platform that can be deployed at scale to assist students in solving non-routine mathematics problem and serve as a blueprint for problem-solving instructions in other domains.

Keywords

Non-routine Mathematics, Student Modeling, Crowdsourcing

1. INTRODUCTION

Training students to be good problem solvers is an important focus of mathematics teaching and learning [9, 25]. However, typical mathematics instruction in U.S. classrooms too often comprises rote learning, such as memorizing the rules for solving algebraic equations, without exposing students to the reasons for those procedures or alternate ways to solve problems [8, 24]. As a result, students struggle to apply learned procedures in novel or non-routine situations where either the surface or structural features of the problem are different [3]. Not surprisingly, U.S. students have fallen far behind their counterparts in other countries on standardized tests of mathematics achievement and on tests measuring abilities to solve novel and challenging problems [18].

Addressing this issue at scale requires developing an educational platform that provides individualized instruction to students in non-routine mathematics problems. However, there are two crucial questions in this direction. First, what do we teach the students? Due to the non-routine domain's nature, there is no set

of procedures that can be used to solve all problems. Thus, we choose to focus on identifying a small number of strategies that are most widely applicable; these will serve as the *expert model*. Second, how do we evaluate student inputs and provide effective feedbacks? Recent development of tutoring systems has moved on to more complicated knowledge domains, such as protein folding [2], programming language [1], and database [17], which also require students to engage in high level problem-solving tasks. To our knowledge, however, there is no system that targets mathematics problem-solving. This domain is unique in that students have to input mathematical arguments, which are a combination of text and formula, to explain and justify their results. Therefore, our approach is to first construct a dataset of mathematical arguments and explanations through crowdsourcing, then employ natural language processing techniques trained on the dataset to assess the quality of student inputs.

2. RESEARCH PLAN

2.1 Constructing the Expert Model

Schoenfeld [21] identified non-routine problems as those: (a) without straightforward, algorithmic solutions, (b) that require ingenuity and resourcefulness to solve, and (c) with multiple solutions. A useful source of these problems is the high school textbook "Problem Solving Strategies: Crossing the River with Dogs" [12] that contains problems such as:

- Each team in a new basketball league will play one game against each of the other teams. There are seven teams: A, B, C, D, E, F, G. How many games will be played in all? (p. 32)
- Find the sum of the first 5000 odd numbers (p. 296)
- How many squares are there on an 8 x 8 grid? Note that a square can have side length from 1 to 8. (p.271)

In identifying a conceptual framework for solving these problems, we turned to the four basic principles of problem-solving outlined by Polya [19]. First, "Understand the problem" is to make sure one knows what is being given and asked. Second, "Devise a plan" involves finding the connection between the data and the unknown, then choosing an appropriate strategy, such as making an orderly list, eliminating possibilities, and looking for a pattern. Third, "Carry out the plan" means executing the chosen strategy. Finally, "Look back" is about reflecting on the problem and the solution so that one can acquire transferable insights for new problems. Guided by these principles, we have solved 70 non-routine mathematics problems from [12], which we will use as the source of instructional materials for this project. Through this exercise, we also came up with eight strategies, four of which are for "preprocessing," used to create a useful mental representation of the problem, and four of which are "solving" strategies, used to come up with a solution. Every one of the 70 problems were solved in multiple ways, using different strategies.

Huy Nguyen, John Stamper and Bruce McLaren "Towards Modeling Students' Problem-solving Skills in Non-routine Mathematics Problems" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 702 - 705

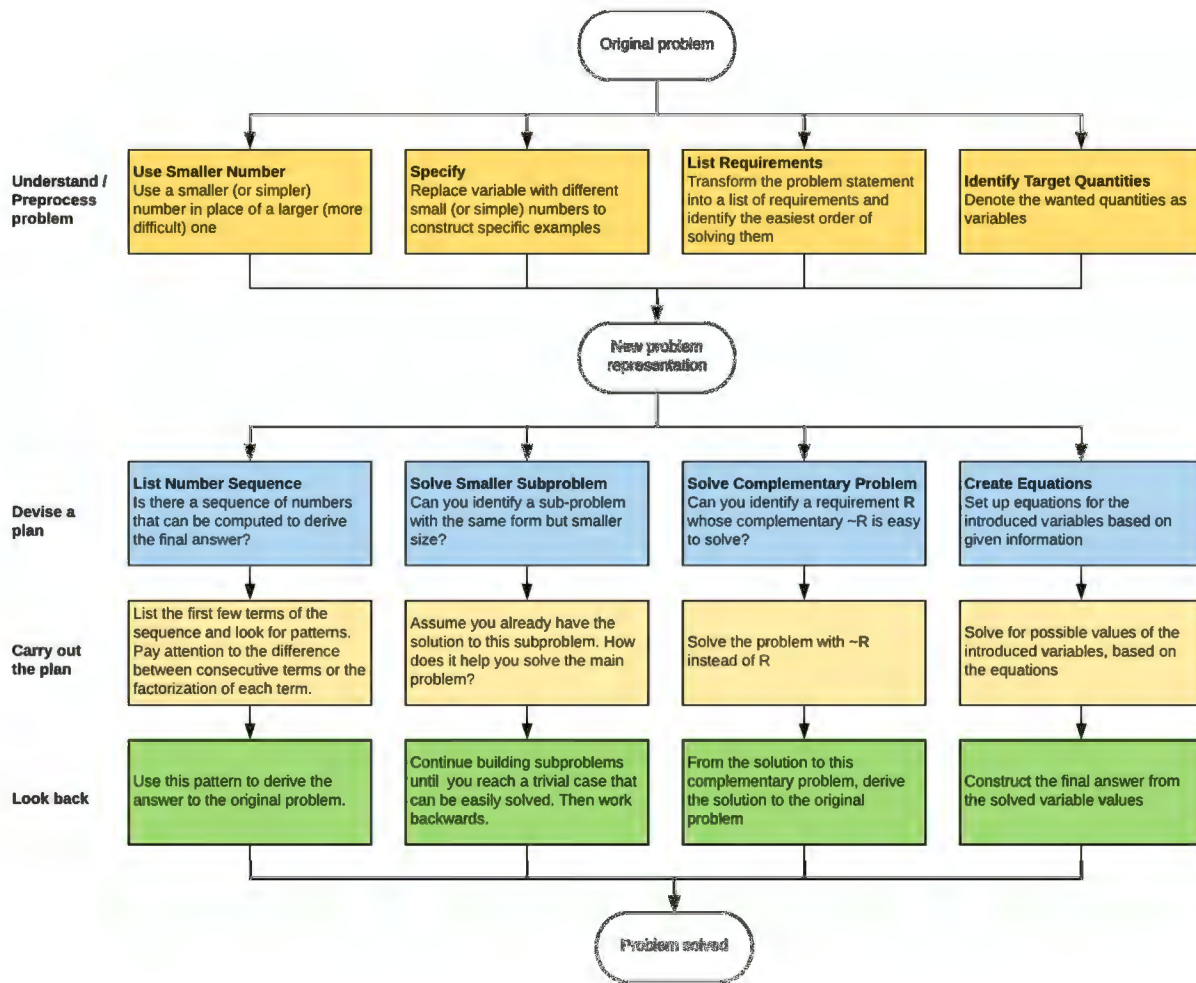


Figure 1: Non-routine problem-solving flowchart serving as the expert model.

Table 1: Materials used in the study.

Problem Statement	Comparison questions
Problem 1: How many squares are there in an 8 x 8 chessboard?	Which solution strategy do you prefer and why? Which solution strategy works better for a larger chessboard? Which solution strategy works better if the question prompt changes to “how many rectangles are there in an 8 x 8 chessboard”?
Problem 2: A number is called a decreasing number if it has two or more digits and each digit is less than the digit to its left. For example, 7,421; 964,310; and 52 are decreasing numbers but 3,241; 6,642 and 963,212 are not. How many decreasing numbers are there?	Which solution strategy do you prefer and why? Which solution strategy would work better if the definition of decreasing number change to “a number is called a decreasing number if it has between two and ten digits, and each digit is less than or equal to the digit to its left”?

To characterize problem-solving strategies -- as well as to create a representation that can be used to guide students -- we developed the flowchart in Figure 1 from solving the 70 problems, organized according to Polya’s four problem-solving stages [19].

The flowchart will serve as a scaffold as students are exposed to and practice different problem-solving strategies. It is intended to

help students select the most appropriate strategy at each stage of the problem-solving process, based on the problem features.

Once the students have selected which strategy to follow, they will be prompted to enter the arguments and explanations for each of the four steps in Figure 1.

2.2 Pilot Experiment

To test the effectiveness of the proposed framework, we will reach out to local high schools to conduct a pilot study, in which students will be shown each of the two non-routine problems in a separate paper and asked to solve it in as many ways as possible, with the aid of the flowchart. When the students succeed in coming up with one solution, they would be asked to think of another solution with a different strategy. If they could not come up with the second solution, they would be prompted to compare a given expert solution with their first solution. If they could not come up with any solution (due to time limit or being stuck), they would be shown one solution strategy, then another. We also rotate the order of solution strategies for each student. Finally, once students have two solutions, they are asked to compare the two with a number of prompting questions. The problem statements, selected from [12], and comparison questions used in the study are listed in Table 1.

Students will be interviewed separately for about an hour and asked to think aloud while solving the problems / evaluating the solutions. Their written solutions and scratch papers, in addition to think-aloud records, will be used for preliminary data analysis and student modeling.

2.3 Constructing the Student Model

An important component of our educational platform will be its capability to assess student explanations generated during problem-solving and providing feedback on and guidance for those explanations. To achieve this, we propose to combine multiple sources of data, both domain-general and domain-specific, to build the training dataset for this task. Our first data source will be domain-general, built from online math problem-solving datasets that include solutions and explanations for math word problems. Three datasets that we will explore are those from the Google DeepMind project [15], the Stanford Natural Language Inference (SNLI) Corpus [4] and e-SNLI, a large corpus of explanations for the SNLI dataset [5]. Our second, domain-specific dataset will be crowdsourced. We will use 100 Amazon Mechanical Turkers to collect explanations for each step of each of 15 non-routine mathematics problems (times 2 solutions for each problem) that we will use in our study. In other words, each of the 100 Turkers will be presented with 30 problems and will be tasked with providing explanations for each of the steps in each solution. We will then manually code these explanations according to their quality. Given the coded data, we will experiment with a variety of deep learning models to classify the quality of student inputs. We will explore several variations of Sequence Models and Convolutional Neural Networks coupled with Transfer Learning, as informed by state-of-the-art research results achieved by Howard and Ruder [10]. They proposed Universal Language Model Fine-tuning that reduced current text classification errors by 18%-24% in tasks such as Sentiment Analysis and Topic Detection. We will also experiment with the question classification method used by [20] that achieved high accuracy. In addition, once we have a fully-fleshed coding scheme for explanations, we will look more deeply at the specific content of feedback, particularly in catering feedback to specific knowledge levels of students. After assessing the quality of student inputs at each step, we can model students' learning by Knowledge Component (KC) based modeling [13], where each KC represents one of the 12 strategy steps in Figure 1. We recognize that the level of granularity of these strategies may not be low enough for the final KC model. To address this issue, can use the students' collective performance in each KC, visualized

by learning curves [7] in DataShop [23], in order to apply the human-machine discovery method [22] and identify potential improvements in our initial KC model (e.g., whether there is a KC that needs to be decomposed). We then can revise instructional materials for the next study iteration to identify more fine-grained KCs [14, 16]. There is also potential in KC-based student performance prediction, using a variety of techniques such as Item Response Theory [11], Bayesian Knowledge Tracing [7] and Bayesian Network [6]. Accurate predictions by the system would then allow for more effective and timely interventions, for example by offering an appropriate level of scaffolding at the beginning of a new problem.

3. CONCLUSION

In this paper, we propose a plan to model high-school students' problem solving skills through an educational platform in the domain of non-routine mathematics problems. We envision three potential applications of this platform. First, it will be accessible to a wide range of students, especially the lower performing students who are not traditionally exposed to problem solving. Second, it will provide a blueprint for automated assessment of students' explanations in an open-ended domain, which is among the central themes of the EDM community. Finally, the constructed student models could yield insights into students' problem-solving process that are applicable to other domains.

4. REFERENCES

- [1] Al-Bastami, B.G. and Naser, S.S.A. 2017. Design and Development of an Intelligent Tutoring System for C# Language.
- [2] Bauer, A. and Popović, Z. 2017. Collaborative problem solving in an open-ended scientific discovery game. *Proceedings of the ACM on human-computer interaction*, 1(CSCW), p.22.
- [3] Bernardo, A. B. 2001. Analogical problem construction and transfer in mathematical problem solving. *Educational Psychology*, 21(2), 137-150.
- [4] Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*. <https://nlp.stanford.edu/projects/snli/>
- [5] Camburu, O. M., Rocktäschel, T., Lukaszewicz, T., & Blunsom, P. (2018). e-SNLI: Natural Language Inference with Natural Language Explanations. In *Advances in Neural Information Processing Systems* (pp. 9560-9572).
- [6] Conati, C., Gertner, A. and Vanlehn, K., 2002. Using Bayesian networks to manage uncertainty in student modeling. *User modeling and user-adapted interaction*, 12(4), pp.371-417.
- [7] Corbett, A.T. and Anderson, J.R., 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4), pp.253-278.
- [8] Crooks, N. M., & Alibali, M. W. 2014. Defining and measuring conceptual knowledge in mathematics. *Developmental Review*, 34(4), 344-377.
- [9] Givvin, K. B., Stigler, J. W., & Thompson, B. J. 2011. What community college developmental mathematics students understand about mathematics, Part II: The interviews. *The MathAMATYC Educator*, 2(3), 4-18.
- [10] Howard, J., & Ruder, S. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th*

Annual Meeting of the Association for Computational Linguistics (Vol. 1, pp. 328-339).

- [11] Johns, J., Mahadevan, S. and Woolf, B., 2006, June. Estimating student proficiency using an item response theory model. In *International Conference on Intelligent Tutoring Systems* (pp. 473-480). Springer, Berlin, Heidelberg.
- [12] Johnson, K. & Herr, T. 2001. Problem solving strategies: Crossing the river with dogs and other mathematical adventures. *Key Curriculum Press*, 2nd edition.
- [13] Koedinger, K.R., Corbett, A.T. and Perfetti, C., 2012. The Knowledge□Learning□Instruction framework: Bridging the science□practice chasm to enhance robust student learning. *Cognitive science*, 36(5), pp.757-798.
- [14] Koedinger, K.R., Stamper, J.C., McLaughlin, E.A. and Nixon, T. 2013. Using data-driven discovery of better student models to improve student learning. In *International Conference on Artificial Intelligence in Education* (pp. 421-430). Springer, Berlin, Heidelberg.
- [15] Ling, W., Yogatama, D., Dyer, C., & Blunsom, P. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.
- [16] Liu, R. and Koedinger, K.R., 2017. Closing the Loop: Automated Data-Driven Cognitive Model Discoveries Lead to Improved Instruction and Learning Gains. *Journal of Educational Data Mining*, 9(1), pp.25-41.
- [17] Mitrovic, A., Ohlsson, S. and Barrow, D.K. 2013. The effect of positive feedback in a constraint-based intelligent tutoring system. *Computers & Education*, 60(1), pp.264-272.
- [18] Mullis, I.V.S., Martin, M.O., and Foy, P. 2008. *TIMSS 2007 international mathematics report: Findings from IEA's trends in international mathematics and science study at the fourth and eighth grades*. Chestnut Hill, MA: TIMSS & PIRLS International Study Center, Lynch School of Education, Boston College.
- [19] Polya, G. (2004). *How to solve it: A new aspect of mathematical method* (No. 246). Princeton university press.
- [20] Ruseti, S., Dascalu, M., Johnson, A. M., Balyan, R., Kopp, K. J., McNamara, D. S., Crossley, S. A., & Trausan-Matu, S. (2018). Predicting question quality using recurrent neural networks. In *Proceedings of the International Conference on Artificial Intelligence in Education (AIED 2018)* (pp. 491-502). Springer, Cham.
- [21] Schoenfeld, A. H. 1983. The wild, wild, wild, wild, wild world of problem solving: A review of sorts. *For the Learning of Mathematics*, 3, 40-47.
- [22] Stamper, J.C. and Koedinger, K.R. 2011. Human-machine student model discovery and improvement using DataShop. In *International Conference on Artificial Intelligence in Education* (pp. 353-360). Springer, Berlin, Heidelberg.
- [23] Stamper, J., Koedinger, K., d Baker, R. S., Skogsholm, A., Leber, B., Rankin, J., & Demi, S. 2010. PSLC DataShop: A data analysis service for the learning science community. In *International Conference on Intelligent Tutoring Systems* (pp. 455-455). Springer, Berlin, Heidelberg.
- [24] Stigler, J., & Hiebert, J. 1997. Understanding and improving classroom mathematics instruction: An overview of the TIMSS video study. In *ACER National Conference 1997* (pp. 52-65).
- [25] Stigler, J. W., Givvin, K. B., & Thompson, B. 2010. What community college developmental mathematics students understand about mathematics. *The MathAMATYC Educator*, 10, 4-16.

Anatomy of mobile learners: Using learning analytics to unveil learning in presence of mobile devices

Varshita Sher
School of Interactive Arts and Technology
Simon Fraser University, Canada
vsher@sfu.ca

ABSTRACT

Mobile technology has been a focus of research since the early 2000s and has attracted researchers from various disciplines ranging from pedagogical, health-care, technological to app developers. In recent times, there has been a substantial interest in capitalizing on the abundance and the ubiquity of these technologies for their educational use. However, the role of mobile phones in educational setting is still largely under-researched. Similarly, little attention has been paid to the research on the extension of learning analytics to analyze the learning processes and strategies of students adopting mobile platforms. Traditionally, the research into mobile learning has mainly relied upon self-reported data. While literature has evidence that survey data facilitates extraction of invaluable information, it suffers from a significant shortcoming - unreliability due to learner bias and poor recall. Therefore, this paper outlines the doctoral research project that explores the use of mobile technology in educational context using data mining techniques and learning analytics methods to analyze digital trace data and provide insights into how students learn. The results so far have enabled us to categorize students as adopting one of the three technological modality strategies - strategic, minimalist, and intensive - based on how extensive the use of multiple modalities (such as desktops, tablets, smartphones) is for learning activities and their final academic performance. Our results also provide evidence suggesting incorporation of the modality used by the learner, for carrying out an activity, as a viable feature in learner models helps in improving the prediction power of these models.

Keywords

Learning Analytics, Mobile Learning, Trace Analysis

1. INTRODUCTION

Mobile technology has been a focus of research since the early 2000s and has attracted researchers from various disciplines ranging from pedagogical, health-care, educational,

technological to app developers. An October 2011 article in *The Economist* posited that, with the number of PCs already surpassing 1 billion in 2008, the number of mobile devices too would reach 10 billion in 2020 [2]. However, even with the proliferation of mobile phones at such an unprecedented rate, their role in the educational setting is still largely under-researched [1, 12]. The challenge for educators and designers, thus, is one of understanding and exploring how best students might use mobile technology to support learning. This challenge is further complicated by the fact that while there exist plethora of learning analytics dashboards (LADs), there is a critical paucity of mobile learning analytics dashboard applications [11] in comparison to their desktop counterpart. The desktop LADs have been extensively researched in terms of their usability, learner strategies, usefulness, effectiveness, and efficiency using complex data mining techniques and learning analytics methods. On the contrary, little attention has been paid to the research on the extension of learning analytics to analyze the learning process of students adopting mobile platforms. The very few extant mobile LADs [5, 10, 7, 4] adopted by students have been analyzed mostly using self-reported data typically collected through questionnaires or think-aloud protocols, which suffer from unreliability issues due to learner bias and poor recall. Moreover, we are still unaware of the impact of the 'source' of log files (from different devices), if any, on the outcome prediction in learner models.

Thus, our research aims to bridge all the previously discussed gaps and explore the use of mobile technology in an educational context. It aims to provide a holistic understanding of learning in presence of mobile devices and its impact on learning - right from the identification of learning strategies employed by mobile learners, to the learning activities benefiting the most from mobile technology (w.r.t. online discussions and course assignments), and to the construction of technological modality-specific learner models for learning outcome prediction. We focus on using advanced data mining techniques and learning analytics methods to analyze digital trace data and provide insights into how students learn using different technological modalities, with main focus on mobile devices.

1.1 Research Question I

The first goal of the proposed research is to explore how mobile devices are used when regulating learning via learning management systems (LMS) in the context of blended courses. To do so, we mine the sequence data from student

Varshita Sher "Anatomy of mobile learners: Using learning analytics to unveil learning in presence of mobile devices" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 706 - 709

logs to examine the extent to which various technological modalities (including mobile devices, laptops and desktop PCs) are either used sequentially and/or simultaneously and assess their potential to influence the overall academic performance and study habits at various learning activities. In order to achieve that, we study the following two research questions:

RQ1.1: *Can we detect patterns in students' use of multiple modalities that are indicative of their adopted technological modality strategy when using an LMS tool? If so, what kind of strategies emerge?*

RQ1.2: *Is there an association of the identified strategies with students' performance in online discussions and overall academic performance?*

1.2 Research Question II

Central to the idea of mobile learning is that learning can occur context-free; across different places and at different times and not confined to the formal classroom settings. While location-aware mobile learning systems has been widely studied, albeit using descriptive statistics only, even lesser attention has been given to the temporal aspect of the use of these mobile technologies. That is to say, not much is known of the associations between different modalities (such as desktops, mobiles and tablets) and the time of the day during which the modality-learner interactions take place. We assess the associations of time of the day they learn, not only with the frequency of usage of a modality, but with the sequential patterns of usage of different modalities in a blended course. Knowledge of the personally-negotiated learning time-frames, mediated by different modalities are conducive to timely, personally tailored feedback, reinforcing the 'right information at right time' learning motto. In order to achieve that, we study the following research question:

RQ2: *Are there any underlying associations between time of the day and the patterns of modality usage based on a learner's modality-use profile?*

1.3 Research Question III

The introduction of mobile technology as a pedagogical tool has witnessed many enthusiastic supporters who successfully incorporate mobility in their everyday learning routine. However, it is still unclear what dictates the students' decision to adopt or resist mobile technology in the first place. The following research questions ascertain whether learners' patterns of modality usage are driven by their inherent student characteristics or the type of activities they must engage in.

RQ3: *Do students use technological modalities differently when engaging with different types of learning activities (say, Assignments and Online Discussions)?*

1.4 Research Question IV

The research area of analyzing log file trace data to build academic performance prediction models has tremendous potential for pedagogical support. Currently, these learner models are developed from logs that are composed of one

intermixed stream of data, treated in the same manner regardless of which platform (mobile, desktops) the data came from. Given that learners use a combination of devices when engaging in learning activities, it is apparent that weighing the logs based on the platform they originate from might generate different (possibly better) models, with varying priority assigned to different model features. For instance, frequency of course material access might be a less powerful indicator of academic performance compared to the frequency of course material access 'from mobile devices', probably due to the benefits associated with ubiquitous any-time access available to mobile learners. Thus, the primary goal of this research question is to bring to light the potential for improvement of prediction power of models after considering the learner's platform of access while generating learner models for predictive analysis.

RQ4: *To what extent is the predictive strength of LMS features influenced by distinguishing the modality of learner access when predicting course grade??*

2. POTENTIAL CONTRIBUTIONS

The research on mobile learning has primarily focused on studying the *effectiveness* or *design* of the mobile learning systems [13]. There are two major flaws to this. Firstly, as pointed out by [6], *it is the learner that is mobile, rather than the technology*. Hence, while the research focused on designing of specific portable technologies has been useful, it is now time to dig deeper into complex interactions between learners, mobile devices, learning activities, and available learning materials (briefly touched upon by [1]).

Furthermore, assessments of effectiveness of mobile learning systems are generally conducted using overly general, broad surveys and self-reported questionnaire, usually in a lab setting. The traditional surveys have been recognized as highly flawed in the educational research community due to unreliable issues stemming from learner bias and poor recall and the extremely controlled environment in lab settings deters the observation of learners' actions, decisions and learning strategy choices in their natural environment (thereby threatening external validity of experiment). Consequently, the studies have only sufficed in making superficial claims about *trends* in mobile learning, using simple analyses such as aggregates and percentages - *20% people prefer to use mobile phones for participation in the discussion activity*. This is insufficient for the explication of the actual way in which mobile technology is impacting the everyday learning process in authentic educational settings wherein interleaved pattern of usage are observed which up until now have been understudied [8, 9]. This is exactly what my doctoral thesis will cater to. The advantages of analyzing such patterns is three-fold. It supports instructors in blended courses through more refined interpretation of students' actions in the LMS when participating in the learning activity. For learners, this allows for recognition of strategies (comprising modality-action pairs) that maximizes student's learning achievements and helps in refraining from suboptimal learning strategies. It is equally useful for future LMS designers when designing notification systems that are capable of sending reminders for modality-specific learning actions.

3. METHOD AND CURRENT PROGRESS

In my research, I am analyzing the data produced by the second and third year undergraduate students in two programming oriented courses at a Canadian university. Currently, the data has been collected from two semesters (Fall 2017 and Spring 2018) and is being continually logged from subsequent offerings of the two courses. Each course spans over 13 weeks and for the two semesters, a combined enrollment of 121 students (83+38) was observed. The courses use blended delivery, utilizing the university's learning management system (LMS) to support learning activities and students' overall schoolwork. The LMS hosts access to reading material, posted lecture slides, tutorial materials, general course information, weekly or bi-weekly course assignments, assignment submission, grades, and allows participation in online discussion activities. In addition to the web-browser versions of the LMS (desktop/laptop/mobile), students have access to the mobile app version provided by the LMS vendor. Comparison of the features and functionalities offered by the two versions have revealed no apparent differences.

I plan to carry out my research in four phases, one for each research question. At the current stage, I am working on phase 2 and 3, having completed and submitted research questions from phase 1 (accepted) and phase 4 (in review). Hence, in this section, I will briefly touch upon the methodology used (or intend to use) for each of these research questions and the results obtained so far.

3.1 RQ1

In order to examine the presence of patterns in students' use of several technological modalities, I encoded each learning session as a sequence of modalities (used to carry out each action in that learning sessions) using a representation format of the TraMineR R package [3]. These sequences were clustered using agglomerative clustering based on Ward's method. The computation of the distance (similarity) between sequences, required for the clustering algorithm, was based on the optimal matching distance metric [3]. The sequence clustering algorithm produced four clusters, i.e. technological-modality profiles - Diverse (use of many different modalities), Mobile-oriented, Short-desktop and Desktop. Next, the students were clustered, based on how many of their sequences belong in each modality profile, using Euclidean metric to compute the distance between vectors. As a result, three student clusters (Strategic, Minimalist and Intensive), representative of their modality strategies, were obtained.

To examine if there was a significant difference between the identified student groups, we performed a multivariate analysis of variance (MANOVA). The student cluster assignment was treated as the single, independent variable along with following dependent variables: overall academic score, counts and time spent on viewing, posting and replying to messages along with the word counts and quality of messages in discussion board. I found a moderate effect size ($\epsilon^2 = 0.12$) of students' adopted strategies on the final course grade. Furthermore, when looking specifically at online discussion engagement and performance, students' adopted technological modality strategies explained a large amount of variance ($\eta^2 = 0.68$) in their engagement and quality of contributions.

3.2 RQ2

For this question, I have encoded learning sessions from all students (as was done in RQ1), followed by categorizing each learning session into one of the four broad TOD (time of the day) categories, intuitively: Morning (5 - 11 a.m.), Afternoon (11 a.m. - 4p.m.), Evening (4 - 7 p.m.) and Night (7 p.m. - 5 a.m.). To examine if there was an overall significant relation between the modality-profile of learning sessions and the time of the day each of these sessions took place, a chi-square test of independence was performed across all learning sessions after summarizing data, composed of each sessions' technological modality profile cluster and the TOD category it belongs to, in a two-way contingency matrix. The analyses was replicated across each of the three modality strategies (obtained in RQ1) and results revealed significant associations for all three strategies. However, there were significant differences with respect to the preferred time of the day for carrying out learning sessions belonging to each modality profile, based on the learner's modality strategy. For instance, mobile-oriented learning sessions were carried out mostly in the afternoon by strategic and intensive learners but in the morning by minimalist learners.

3.3 RQ3

The ongoing activities for this research question are mainly focused on preliminary data analysis. I have encoded the learning sequences, from two main learning activities i.e. assignment and online discussions, as representations of the TraMineR format. For each learning activity, the sequence clustering based on optimal matching metric resulted in four and three technological modality profiles (TMP), respectively. Consequently, two different partitioning of the students, one based on TMP clusters from online discussion activity and the other based on TMP clusters from assignment activity, both resulted in 3 student clusters each.

The initial exploratory analysis comparing the two student clusterings, obtained from the two learning activities, revealed a rand index of 0.48, meaning that the two clustering agree to a very small extent only. Therefore, we posit with some certainty that the learner's patterns of modality usage (or in short their choice of technological modality strategy) is dependent on the learning activity they are engaging in. To further strengthen this claim, we look at the rand indices obtained from the two learning activities when compared to the benchmark student classification (from RQ1). The benchmark classification corresponds to the student strategies (i.e. clusters) that are gauged from their overall engagement with the LMS and reflect the *generic* or habitual patterns of use of different modalities (and thus considered analogous to a student's innate characteristics). We found a large overlap (rand index = 0.85) of strategies in the benchmark classification with the assignment activity but only a small overlap (rand index = 0.51) with online discussion activity. This indicates that, in addition to the strategies from the two learning activities being different, the strategies from one of them (i.e. assignment activity) *more* closely resemble the strategies in the benchmark classification, compared to the other (i.e. discussion activity). As with any speculation, follow up inferential analysis will be required to solidify our claims.

3.4 RQ4

To investigate the effect of modality on different types of commonly included learning-related activities (Table 1) and their traces in the online courses, I selected 10 features (5 counts + 5 time spents for each activity) for inclusion in my analyses as predictors of academic success. Variables derived from the LMS trace data include: syllabus, course material (lecture + tutorial slides and instructor provided supplementary material), assignments, feedback on the assignments and calendar. For each student I extracted the number of times (and the time spent on) using a particular feature by aggregating individual operations. I call these variables *LMS features*. Each of these variables was split up further to account for the platform used to access that particular feature. For instance, in addition to having the total number of assignment views for a student, I computed three more variables – mobile views, desktop views and tablet views – which indicate the respective number of assignment views from each of the three main modalities. I call such variables *Modality features*.

Next, I conducted a series of regression analyses with course grade as the outcome variable in each. For each of the ten learning features, two regression models were built using (a) LMS features and (b) Modality features. The results reflected that for each of the ten features, an increase in R^2 from Model 1 to Model 2 was observed. This increase accounts for the percentage of variability in student course grade explained by the Modality features over and above the LMS feature. An ANOVA analysis using F-test of the statistical significance was conducted to ascertain whether the increase was statistically significant and it was found to be significant for more than 50% of the variables tested.

4. ADVICE SOUGHT

My research is at an intermediate stage and I hope the consortium could provide me some guidance and insights in the following areas:

- I am interested to know if there is a method, especially in the area of network analysis that can help me assess the (direction and frequency of) transitions between mobiles, desktops and tablets, and how these transitions differ when learners are engaging in different learning activities.
- For RQ3, while I am focusing on chi-square test of independence to test associations between time of the day and modality patterns, for the next steps I need to know how to identifying/interpret patterns using the time-series analysis. Also, how can I account for the ‘random noise’ that might be introduced from students’ use of different devices for non-educational purposes?

5. ACKNOWLEDGMENTS

This research was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and Social Sciences and Humanities Research Council of Canada (SSHRC).

6. REFERENCES

- [1] N. R. Aljohani and H. C. Davis. Significance of learning analytics in enhancing the mobile and pervasive learning environments. In *Next Generation Mobile Applications, Services and Technologies (NGMAST), 2012 6th International Conference on*, pages 70–74. IEEE, 2012.
- [2] T. Economist. Beyond the pc, Oct 2011.
- [3] A. Gabadinho, G. Ritschard, N. S. Mueller, and M. Studer. Analyzing and visualizing state sequences in r with traminer. *Journal of Statistical Software*, 40(4):1–37, 2011.
- [4] H. Heflin, J. Shewmaker, and J. Nguyen. Impact of mobile technology on student attitudes, engagement, and learning. *Computers & Education*, 107:91–99, 2017.
- [5] J. Nakahara, S. Hisamatsu, K. Yaegashi, and Y. Yamauchi. itree: Does the mobile phone encourage learners to be more involved in collaborative learning? In *Proceedings of the 2005 conference on Computer support for collaborative learning: learning 2005: the next 10 years!*, pages 470–478. International Society of the Learning Sciences, 2005.
- [6] M. Sharples, J. Taylor, and G. Vavoula. Towards a theory of mobile learning. In *Proceedings of mLearn*, volume 1, pages 1–9, 2005.
- [7] R. Shen, M. Wang, and X. Pan. Increasing interactivity in blended classrooms through a cutting-edge mobile learning system. *British Journal of Educational Technology*, 39(6):1073–1086, 2008.
- [8] G. Stockwell. Using mobile phones for vocabulary activities: Examining the effect of platform. 2010.
- [9] G. Stockwell. Tracking learner usage of mobile phones for language learning outside of the classroom. *CALICO Journal*, 30(1):118–136, 2013.
- [10] B. Tabuenca, M. Kalz, H. Drachsler, and M. Specht. Time will tell: The role of mobile learning analytics in self-regulated learning. *Computers & Education*, 89:53–74, 2015.
- [11] K. Verbert, S. Govaerts, E. Duval, J. L. Santos, F. Van Assche, G. Parra, and J. Klerkx. Learning dashboards: an overview and future research opportunities. *Personal and Ubiquitous Computing*, 18(6):1499–1514, 2014.
- [12] I. S. H. Wai, S. S. Y. Ng, D. K. Chiu, K. K. Ho, and P. Lo. Exploring undergraduate students’ usage pattern of mobile apps for education. *Journal of Librarianship and Information Science*, page 0961000616662699, 2016.
- [13] W.-H. Wu, Y.-C. J. Wu, C.-Y. Chen, H.-Y. Kao, C.-H. Lin, and S.-H. Huang. Review of trends from mobile learning studies: A meta-analysis. *Computers & Education*, 59(2):817–827, 2012.

Modeling Student Performance and Disengagement Using Decomposition of Response Time Data

Deniz Sonmez Unal
School of Computing and Information
University of Pittsburgh
Pittsburgh, PA
des204@pitt.edu

ABSTRACT

Students' first response time has been used as a key predictor in various types of models in order to predict student performance and engagement. However, the effect of how much time students spend on particular activities within a single response time remained unexplored. In this paper, we divide response times into meaningful subcategories in order to provide a more accurate prediction of learning and disengagement. We fit a model with the derived subcategories and show how this model outperforms the baseline model that uses the raw response times. We present the early findings of our ongoing work and seek advice on how the insight that we get from these findings can be used in further detection of student off-task behavior.

Keywords

Student modeling, Response time, Disengagement, Regression models

1. INTRODUCTION

Various types of models of student learning from intelligent tutoring systems leverage response times in order to better predict student performance. For example, Xiong et al. [10] compared a baseline random forest model to another model with response times and related features added. They found that adding response time could make a (small but) significant improvement in predicting student performance. Wang et al. [9] included response time information in knowledge tracing algorithm and found that adding response time improved KT's accuracy. People have also used response times in their models to explain whether students are engaged with the system they are interacting. Beck developed a model [2] based on item response theory which was able to give an estimate of the probability of a correct response given a response time for modeling disengagement over time. Shih et al. [7] developed a model that shows some behaviors previously shown as negatively correlated with learning can actually be beneficial for some students. In general, much of this research is based on the hypothesis that if a student responds to a problem step too quickly or too slowly, they are likely to be unsuccessful in that particular step and if students do this too frequently, they tend to perform worse on the task.

However, something that is less explored is how much time students are spending on different activities within a single response time and how it is affecting how they perform on tasks. If we take a reading task for example, we would need an estimate of how long it takes a student to read a given text so that we could tell when a student is switching from one engaged state (reading) to another (thinking about the step/reflection) or an off-task/disengaged state. Shih's model used estimates of unobserved activities during a single response time to show that repeatedly asking for hints might not always be a harmful behavior. Our work is similar to Shih's work in a way we are trying to estimate how much time students spend on activities we cannot directly observe from the log data, but our activities of interest are different as Shih is focusing on activities can be observed in a problem solving domain, while we are interested in a reading task.

The goal of our work is to decompose response times into meaningful subcategories in order to provide a more accurate prediction of learning and disengagement.

2. PROPOSED CONTRIBUTIONS

In this work, we propose to develop a new model that leverages time information to predict student performance and disengagement. In order to achieve this goal, we will use the log data collected from EMBRACE [8]. EMBRACE is an iPad application designed to improve English reading comprehension of young native Spanish speaker students. Students are reading interactive story books by chapters and answer multiple choice questions related to the story at the end of each chapter. While students are reading, some of the sentences are highlighted with a blue color. In those sentences, students are asked to move images shown on the screen to represent what they read (see Figure 1). The control version of EMBRACE does not have this feature. In this version, student actions are restricted to reading a sentence, tapping on a word to hear its pronunciation, and moving forward to see the next sentence. We start to analyze the response time from the control version. Since the student actions are limited, we can get a clearer idea about how predictive the subcategories of response times we derive. As the students can be either reading the sentence or thinking about it (assuming they are engaged with the system) in the control version, we initially decompose response time into **reading time** and **thinking time**. Reading time is the estimated time for a student to read the sentence and thinking time is the remaining time when we subtract reading time from the raw response time (the total time spent on a sentence).

Deniz Sonmez Unal "Modeling Student Performance and Disengagement Using Decomposition of Response Time Data" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 710 - 713



Figure 1. Screenshot from EMBRACE

However, the student can be in a disengaged or an off-task state during reading. If that is the case, some part of what we identify as thinking time will be actually the time spent in these unproductive states. We are interested in finding in what portion of thinking time students are in these states. Since often being in these states are found to be negatively correlated with learning [1], the student performance within periods of thinking time may be indicative of their cognitive state. Our proposed approach to detect these states is to estimate the break points where the relationship between thinking time and the student performance changes. We are targeting identifying these states based on the different relationships in the regions that are defined by the estimated break points and frequency of specific student behavior such as help requests or trying to skip through sentences.

With the insight gained from the control version, our next step will be to leverage this approach to identify different student behaviors within response times to more complex actions in other versions of EMBRACE. We are aiming to provide better predictions of student performance and identify student cognitive states such as reflection, gaming the system, mind wandering, and wheel spinning based on the time information in a larger set of student actions.

3. METHODS & PRELIMINARY RESULTS

3.1 Data

Our data comes from the logs collected from 22 students interacting with the control version of EMBRACE. Students are reading 35 chapters and answering questions at the end of each chapter. Our measure of student performance is the proportion of correctly answered questions in the comprehension assessment in each chapter. We extract the time a student spends on each sentence in a chapter and calculate the mean response time in each chapter. Some students had unrealistic response times (due to logging errors) or missing assessment scores in some of the chapters, such student chapter pairs are excluded from the dataset. This left us 742 data points (student chapter pairs).

3.2 Estimating Likely Reading Time

Since the actual time that students take to read a sentence is not logged in the application, we estimate how long it should take them based on students' decoding ability and how many words a particular sentence contains.

The reading time was estimated for each sentence as word count in the sentence divided by how many words the student should be reading per minute based on their decoding ability. Students' decoding ability is measured by their scores on the decoding part of the Qualitative Reading Inventory (QRI) [3].

In order to get the words per minute for each student, we used the guidelines from [4]. This gives lower and upper bounds for how many words a student should be reading per minute based on what grade the student is in.

Instead of making predictions about reading time based on grade, we make predictions about reading time based on QRI scores. We perform this mapping as follows:

$$f(t) = c + \left(\frac{d - c}{b - a} \right) * (t - a)$$

where interval $[a, b]$ denotes our QRI range, $[c, d]$ denotes the wpm range for students from 2nd to 4th grade in [4], and t is a specific QRI score.

Our QRI range is $[10, 40]$ and the words per minute range that we picked from Morris's guideline is $[80, 150]$. Based on our mapping function, if a student got 33 on QRI, they are estimated to be reading $80 + ((70)/(30)) * (33 - 10) \sim 134$ words per minute. Using the same function, if a student got 10 on QRI their wpm estimation is 80 and if a student got 40 on QRI their wpm estimation is 150.

3.3 Estimating Likely Thinking Time

We simply calculated the thinking time by subtracting the estimated reading time from the response time logged in the application. If we call the sentence time ST , where ST is the mean time spent on sentences in a chapter and call estimated reading time ERT where ERT is the mean of reading time for a student in a chapter based on student's QRI score and the word count in a sentence. Then thinking time TT is calculated as follows:

$$TT = ST - ERT$$

If a student requested help (tapped on a word to hear its pronunciation) on a sentence, we subtract the time the student spends on listening to the help audio (help time) from thinking time. Help time is calculated as number of help requests on a sentence multiplied by two, as help is a single word read to the student twice, which roughly equates to two seconds.

3.4 Predicting Student Performance

We built a piecewise linear regression with students' mean thinking time on chapters, mean frequency of help requests, mean frequency of attempt to skip, and students' pre-test scores on different English language proficiency assessments (SELPS, QRI, Gates) as predictors and students' correct answer proportion on comprehension assessment tests given at the end of each chapter as the outcome variable.

We used the 'segmented' function in R [5, 6] to build the piecewise model. It takes potential breakpoints for one or more predictors as an input, and fits a piecewise linear function based on those breakpoints (each region, as denoted by two adjacent breakpoints, is fit with a linear regression). The function iterates over possible breakpoints, starting from the user-input breakpoints, until it finds an optimal set of breakpoints.

We decided on three breakpoints for the thinking time variable as we expect to see four different relationships and four different states that can be identified by these relationships: 1) Probably

gaming / Not taking enough time to think, 2) Reflection / Thinking deeply about the sentence, 3) Wheel-spinning / Struggling to understand the sentence, 4) Mind-wandering / Possibly thinking about something else other than the sentence. We picked the initial breakpoints using mean and standard deviation of thinking time. Our first break point is one standard deviation below the mean (-1.95 s), our second break point is the mean thinking time (3.99 s), and our third break point is one standard deviation above the mean (9.93 s).

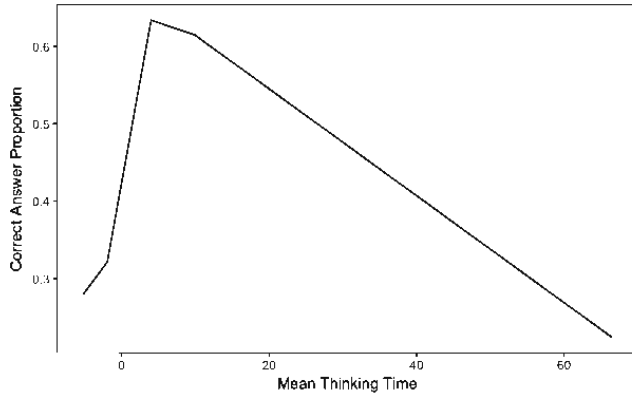


Figure 2. Relationship Between Students' Mean Thinking Time and Correct Answer Proportion

The segmented function estimated the break points at -1.41 s, 4.30 s, and 14.26 s. The resulting segmented model accounted for 21% of the variance ($R^2 = .21$, $F(12, 704) = 16.43$, $p < .01$). The results showed that thinking time ($\beta = -1.36$, $p < .05$), mean frequency of skip attempts ($\beta = -.13$, $p < .001$), English SELPS scores ($\beta = .19$, $p < .001$), and Gates scores ($\beta = .17$, $p < .001$) significantly predicted correct answer proportions.

As a comparison point, we fitted a linear model with the same predictors and the same outcome variable. The resulting linear model could account for 14% of the variance ($R^2 = .14$, $F(6, 710) = 20.13$, $p < .01$). It was found that thinking time ($\beta = .16$, $p < .001$), mean frequency of skip attempts ($\beta = -.16$, $p < .001$), English SELPS scores ($\beta = .18$, $p < .001$), and Gates scores ($\beta = .13$, $p < .01$) significantly predicted correct answer proportions. To statistically compare the linear model to the segmented model we used 'anova' function (which does an F-test) in R. Table 1 shows including the segments in thinking time variable reduces the residual sum of squares significantly which is indicating that the segmented model is a better fit for the data.

Table 1. Comparison of Linear and Segmented Models with Thinking Time

Model	Res. Df	RSS	Df	Sum of Sq	F	Pr(>F)
Linear Model	710	51.408				
Segmented Model	704	46.988	6	4.4199	11.037	8.859e-12

In order to see if using thinking time instead of the raw response time variable extracted from the log data can provide a better fit, we built another piecewise regression model similar to our first model. Instead of thinking time this model uses raw response time as a predictor and estimates the break points on this variable by using its mean and standard deviation values. This model accounted

for 18% of the variance ($R^2 = .18$, $F(12, 704) = 13.32$, $p < .01$). The raw response time ($\beta = -7.08$, $p < .05$), mean frequency of skip attempts ($\beta = -.18$, $p < .001$), English SELPS scores ($\beta = .19$, $p < .001$), and Gates scores ($\beta = .16$, $p < .001$) found to be significant predictors of correct answer proportions. Given this, the piecewise model with the thinking time as the segmented variable gave the best overall fit. In order to compare the two segmented models, 'coxtest' function in R was used since these models are not nested. We found that the fit of the model using thinking time as the segmented variable is significantly better than the one using raw response times ($p < .001$).

Figure 2 shows how the relationship between mean thinking time and correct answer proportion is changing at these points. The relationship seems to match our expectations. In the first zone, students are spending less than a second on thinking which means that they are spending less time than we estimated on reading the sentences, too. This was interpreted as possible gaming behavior as very low time spent on sentences with very low performance on chapter can be sign of skipping the sentences without reading. In the second zone we see that with more time spent on thinking about the sentences, the student performance is reaching to its peak value. This relationship was interpreted as a sign of reflection, taking enough time to think about the sentences after reading which is helpful for better understanding the story and doing well on the comprehension assessment test in the end of the chapter. We interpret the third zone as the possible start of wheel spinning. Since student performance is not low, we anticipate that the students in this zone are not completely off-task however as thinking time increases, the performance starts to slightly decrease which means that students are spending more time on thinking than they should. This was interpreted as students starting to struggle in the sentences. The last zone was interpreted as possible mind-wandering as the time spent on thinking is quite high and as it increases the performance decreases drastically. This relationship explains that the students are not involving in task related thoughts.

Our immediate next steps include calculating the percentage of time spent in each region defined by the estimated break points for each student and seeing if there is a correlation between that and behaviors that are indicative of gaming, mind wandering or wheel spinning such as constantly trying to skip through sentences and repeated help requests.

Later we aim to apply a similar methodology in the other versions of EMBRACE where students can perform more complicated actions. We are interested in finding out different possible subcategories of response time and how they can be used to model student behavior in the richer data.

4. ADVICE SOUGHT

For this doctoral consortium, we would like advice regarding the following concerns:

- 1) What are meaningful sub-categories of response time when student actions are not limited to reading only but they are actually interacting with the application by moving images they see step by step?
- 2) In what ways can the relationship between thinking time and student performance be interpreted? How can it relate to student disengagement or off task behaviors and how can this relationship be validated?
- 3) What suggestions in general do you have for the methods we are using to find sub-categories within response time.

Your feedback on these three aspects of our research will be very insightful for us as we are continuing our work with the data from other EMBRACE versions with more complex student actions.

5. ACKNOWLEDGEMENTS

I would like to express my sincere thanks to Erin Walker for her guidance and valuable comments. I also would like to thank M. Adelaida Restrepo, Arthur Glenberg, and Ligia Gomez for useful discussions and their support. This paper is based upon work supported by the National Science Foundation Grant # 1324807.

6. REFERENCES

- [1] Baker, R. S., Corbett, A. T., Koedinger, K. R., & Wagner, A. Z. 2004. *Off-task behavior in the cognitive tutor classroom*. Proceedings of the 2004 Conference on Human Factors in Computing Systems - CHI '04, 383–390. DOI= <https://doi.org/10.1145/985692.985741>.
- [2] Beck, J. 2005. *Engagement tracing: using response times to model student disengagement*. Proceedings of the 12th International Conference in Artificial Intelligence in Education, 88–95. DOI= <https://doi.org/10.1017/S0016756800061069>.
- [3] Leslie, L., & Caldwell, J. S. 2010. *Qualitative Reading Inventory* (5th ed.). New York, NY: Pearson.
- [4] Morris, D. 2008. *Diagnosis and correction of reading problems*. New York: Guilford.
- [5] Muggeo, Vito M. R. 2003. *Estimating regression models with unknown break-points*. Statistics in Medicine, 22, 3055–3071.
- [6] Muggeo, Vito M. R. 2008. *segmented: an R Package to Fit Regression Models with Broken-Line Relationships*. R News, 8/1, 20–25. URL <https://cran.r-project.org/doc/Rnews/>.
- [7] Shih, B., Koedinger, K. R., & Scheines, R. (2010). A response-time model for bottom-out hints as worked examples. Handbook of Educational Data Mining, 201–212. DOI= <https://doi.org/10.1201/b10274>.
- [8] Walker, E., Adams, A., Restrepo, M. A., Fialko, S., & Glenberg, A. M. 2017. *When (and how) interacting with technology-enhanced storybooks helps dual language learners*. Translational Issues in Psychological Science, 3(1), 66–79. DOI= <https://doi.org/10.1037/tps0000100>.
- [9] Wang, Y., & Heffernan, N. 2012. *Leveraging First Response Time into the Knowledge Tracing Model*. Proceedings of the 5th International Conference on Educational Data Mining.
- [10] Xiong, X., Pardos, Z. a, & Neil, T. 2011. *An Analysis of Response Time Data for Improving Student Performance Prediction*. Proceedings of the KDD 2011 Workshop: Knowledge Discovery in Educational Data.

Techniques for Automatically Evaluating Machine-Generated Questions

Zichao Wang
Rice University
Houston, TX 77005
jzwang@rice.edu

ABSTRACT

Recent work [16] has demonstrated the superior quality of questions generated by state-of-the-art neural question generation algorithms. However, evaluating the quality of these questions typically requires a domain expert which is labor intensive and costly. We therefore seek an reliable and *automatic* method to evaluate the quality of the questions. To this end, we propose a simple evaluation criterion, NLL-QA, to replace the negative log likelihood (NLL) criterion universally employed in state-of-the-art neural question generation models to evaluate and select the final output question. Preliminary results demonstrate that, compared to the traditional negative log likelihood criterion, NLL-QA enables neural question generation models to select better questions that are more relevant to the input answer without sacrificing grammatical correctness.

1. INTRODUCTION

We focus on a particular class of neural question generation (NQG) models that takes as input *an answer* and *an answer context*, usually a sentence or paragraph containing relevant information about the answer, and aims to output *a question sentence*. These NQG models [3, 20, 19] are able to generate questions¹ of much higher quality than previous rule-based systems [5, 1, 8], demonstrating their potential in improving performances of machine comprehension and question answering tasks [18, 14, 4] as well as automating the generation of quiz questions in large-scale educational settings [10, 9]. The question generation process using a NQG model is as follows First, beam search samples a list of candidate questions from the model. Then, a post-processing step selects the most probable question according to the negative log likelihood scores (NLL) of the candidate questions as the final output question.

For a question generated in the above manner to be of practical value, it needs to be, at a minimum, free of grammatical errors and relevant to the input answer from which it is generated. Unfortunately, state-of-the-art neural question generation models often

¹In our context, *generate* a question is equivalent to *output* or *select* a question.

Zichao Wang "Techniques for Automatically Evaluating Machine-Authored Homework Questions" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 714 - 717

Context 1: It stands on the Vistula River in east-central Poland, *roughly 260 kilometres (160 mi) from the Baltic Sea and 300 kilometres (190 mi) from the Carpathian Mountains.*

Question 1: How long is the Vistula River in the Vistula River? (10.21)

Question 2: How far is the Vistula River from the Baltic Sea? (10.67)

Context 2: *For exercise*, Tesla walked between 8 to 10 miles per day.

Question 1: How did Tesla travel to 10 miles per day? (12.55)

Question 2: Tesla walked between 8 to 10 miles for what? (14.15)

Table 1: Two examples where the most probable question (Question 1, which is the final output selected by NLL) is unsatisfactory: in Context 1 it contains grammatical errors, and in Context 2 it asks a question irrelevant to the input answer (*italic and underlined text* in Context). In both examples, a less probable question (Question 2) is significantly better than Question 1. The raw negative log likelihood is shown after each question.

generate questions that do not satisfy these two desired properties. Interestingly, we have discovered, by manually examining numerous generated questions, that a less probable question is of much higher quality than the most probable one in a number of cases. To illustrate, Table 1 shows two less probable questions that are more fluent and more relevant to the input answers than the most probable ones. This observation suggests that, even though a satisfactory output question may already appear in the beam search results, the NLL criterion is often unable to select it.

Moreover, we notice that smart changes in the post-processing step, in addition to the model itself, can bring significant benefit in language generation tasks. For example, Google's neural machine translation system augments NLL with length and coverage penalties, considerably improving the BLEU score of the translations [17]. We are thus motivated to explore the possibility of improving the quality of generated questions by redesigning the question selection criterion in the post-processing step of the question generation process.

Contributions. We propose NLL-QA, a novel question selection criterion that combines NLL with a *answer relevance score (QA)* that explicitly evaluates whether a candidate question can be correctly answered by the input answer from which the question is generated. This combined score is then used in place of NLL to select the final output question from the candidate questions output by beam search. Intuitively, this selection criterion encourages the final output question to be relevant to the input answer, an important aspect of a good generated question that NLL fails to adequately evaluate.

We demonstrate via qualitative, quantitative, and human evaluations that NLL-QA selects questions that are more relevant to the input answers than those selected by NLL without sacrificing grammatical correctness. A further advantage of our proposed criterion is its versatility and ease of implementation. Since our criterion is used only in the post-processing step, it does not affect the NQG model itself in any way, and thus can be easily incorporated together with any advances in the NQG model to jointly improve the quality of generated questions. Our work therefore provides a new approach to boost performance of either existing or newly developed NQG models.

2. THE NLL-QA CRITERION

The proposed NLL-QA criterion calculates a composite score S for each candidate question as a convex combination of negative log likelihood and an answer relevance score QA:

$$S = \alpha \text{NLL} + (1 - \alpha) \text{QA}, \quad (1)$$

where $\alpha \in [0, 1]$ is a tunable hyper-parameter. Intuitively, α controls the relative importance of NLL and QA in selecting the final output question. We note that the case of $\alpha = 1$ corresponds to the usual NLL criterion. We have experimented with a range of α values and have observed that setting $\alpha = 0.3$ in Equation 1 worked well.

Answer Relevance Score. To calculate the answer relevance score, we first use a *question answering model* to answer the generated question. The question answering process takes as input a question and a context, and outputs a list of N predicted answers, with a confidence score in range $[0, 1]$ associated with each predicted answer. We then compare the answer predicted by the question answering model to the input answer from which the question is generated. Intuitively, if the question can be correctly answered, i.e., the input and predicted answers are identical or sufficiently similar, we would assign a high answer relevance score to that generated question.

Although automatic question answering is by itself a difficult task, recent question answering models have achieved performance almost on par with humans [2, 13, 15] on benchmark datasets such as SQuAD [12]. Question answering models thus provide a reasonable evaluation of answer relevance of a question without resorting to human evaluators. The answer relevance score QA is calculated as follows:

$$\text{QA} = \min_j (-\log(c_j \cdot \text{sim}(A, A'_j) + \epsilon)), \quad (2)$$

where A is the input answer from which the question is generated, A'_j is the question answering model predicted answer indexed by j , c_j is the confidence score associated with the j th predicted answer, and ϵ is a small number for numerical stability. The similarity function $\text{sim}(\cdot, \cdot)$ outputs a similarity score in range $[0, 1]$ by comparing the input answer to the j th predicted answer A'_j . We use the Ratcliff-Obershelp algorithm for the similarity function in our experiments, which effectively calculates the string similarity between the input and predicted answers.

3. EXPERIMENTS AND RESULTS

We now describe our experiments to evaluate the efficacy of our proposed NLL-QA criterion using qualitative, quantitative and human evaluation metrics.

Model and Dataset. To show the wide applicability of NLL-QA,

Context 1: The 2010 United States Census reported that Fresno had a population of 494,665.

Human: What was Fresno’s population in 2010?

NLL: What was the population of the 2010 United States Census?

Ours: What was the population of Fresno in 2010?

Context 2: In July 1968, ABC continued its acquisitions in the amusement parks sector with the opening of ABC Marine World in *Redwood City, California*; ...

Human: Where was ABC Marine World opened?

NLL: Where was ABC Marine World World World World World World World demolished ?

Ours: Where was the ABC Marine World sector located?

Context 3: The capability of the command module’s *heat shield* to survive a trans-lunar reentry was demonstrated by using the service module engine to ram it into the atmosphere at higher than the usual earth-orbital reentry speed.

Human: What was one thing that was specifically tested on the Apollo 4 test launch regarding the cm?

NLL: What led to the command module?

Ours: What did the command module provide to survive a trans-lunar reentry?

Context 4: *The sexual ethics task force of the United Methodist Church* states that “research shows it [pornography] is not an ‘innocent activity.’”

Human: Which task force states that pornography is harmful?

NLL: What states that “research shows it is an innocent activity”?

Ours: What states that “research shows it is not an innocent activity”?

Table 2: Representative questions selected by different criteria and human generated reference questions. The italic and underlined text in each context is the true answer. NLL+QA consistently selects higher quality questions than NLL.

we implement, test and compare it with NLL on two state-of-the-art NQG models: **LSTM + attention**, which is a sequence-to-sequence LSTM network with attention mechanism [3]² and **LSTM + attention + copy**, which is LSTM + attention model with additionally copy mechanism [20]. We train each model on the SQuAD dataset [12], each entry in which contains an answer, an answer context, and a human generated reference question. During evaluation, for each entry in the SQuAD test set, beam search first samples 10 candidate questions from the chosen NQG model. Then, the two criteria, NLL-QA and NLL, respectively select a top question out of the 10 candidates as the final output.

We use a trained DrQA model [2] to output predicted answers for Equation 2.

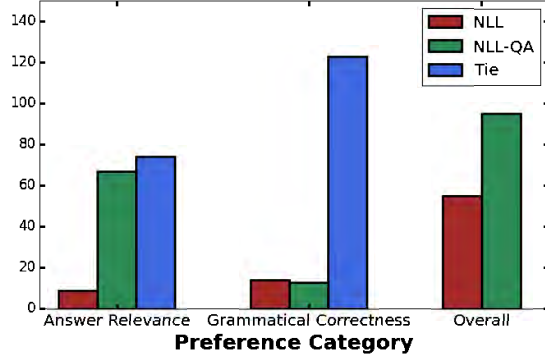
Qualitative Evaluation. For illustration purpose, Table 2 shows representative examples where NLL and NLL-QA return different questions and compare them side-by-side, using LSTM + attention + copy model.³ This allows to observe general differences in performance for these two criteria. It is clear that our combined criterion selects higher quality questions. For example, in Context 2, NLL selects a question that repeats the word “World”, while NLL-QA selects a question that is grammatically correct. In Context 4, both questions are fluent, but NLL selects the question without the crucial negating word “not”, which makes the question inappropriate. On the contrary, NLL-QA correctly selects the question with the appropriate negation.

²To be precise, we modified the implementation of [3] so that it can take the answer word(s) as input.

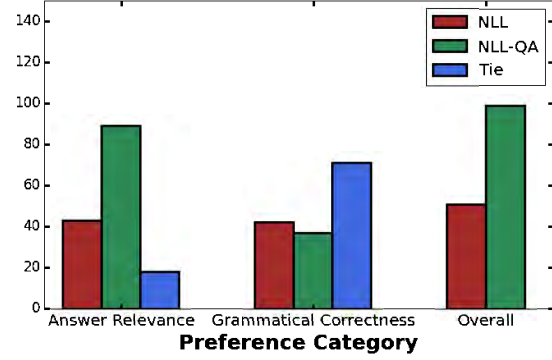
³In our experiments, NLL+QA selects the same questions as NLL in 38.6% of all tests.

NQG Model	Selection criterion	Metrics					
		BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
LSTM + attention	NLL	0.4216	0.2603	0.1797	0.1284	0.1702	0.4212
	NLL-QA	0.4340	0.2684	0.1850	0.1323	0.1766	0.4180
LSTM + attention + copy	NLL	0.4583	0.3030	0.2216	0.1684	0.2035	0.4596
	NLL-QA	0.4661	0.3038	0.2190	0.1637	0.2049	0.4572

Table 3: A comparison between questions selected by NLL and NLL-QA, with the higher scores bold. NLL-QA selects questions of comparable or slightly higher quality for all metrics for both NQG models.



(a) Results for LSTM + attention model.



(b) Results for LSTM + attention + copy model.

Figure 1: Graphical summary of human evaluation results. “Ties” means that raters choose the same preference for both questions, e.g., when raters think both NLL and NLL-QA selected questions are grammatical. NLL-QA dominates in answer relevance and in overall preference. Best viewed in color.

Automatic Evaluation. Similar to prior works [3, 20], we include evaluation results using an array of common automatic metrics including BLEU [11], METEOR [6], and ROUGE [7]. Table 3 summarizes these evaluation results, with the best criterion in bold, for both NQG models. Results in Table 3 show that, overall, NLL-QA selects questions that are of comparable or even slightly higher quality than those selected by NLL. We emphasize that the above evaluation metrics are *not* the most appropriate ones for evaluating the quality of questions, which motivates the human evaluation experiments that we now describe.

Human Evaluation To verify that NLL-QA indeed outputs questions of higher quality than NLL, we conduct a human evaluation experiment for each NQG model. We randomly sample 150 entries from the SQuAD test set where NLL and NLL-QA criteria return different results. For each NQG model, and for each pair of questions, we ask six human raters to give three categories of preferences, one based on grammatical correctness, one on relevance to input answer, and one on overall preference.

We then count the number of times each criterion the raters chose for each of the three preference categories. Figure 1 presents these results graphically for both NQG models. Comparing the last two bars in each plot, we see that, overall, human raters prefer NLL-QA twice as often as NLL. Comparing answer relevance in each plot, we see that human raters strongly prefer NLL-QA than NLL. We performed binomial tests confirm the significance of these two observations: $p = 6.8 \times 10^{-4}$ and $p < 10^{-4}$ for overall preference, and $p = 1.5 \times 10^{-11}$ and $p = 0.01$ for answer relevance, for LSTM + attention and LSTM + attention + copy model, respectively. Comparing grammatical correctness in each plot, there seems to be a slight preference for NLL over NLL-QA. However,

binomial test shows that this slight preference is not statistically significant at all, with $p = 1$ for LSTM + attention model and $p = 0.75$ for LSTM + attention + copy model.⁴ We thus conclude that NLL-QA significantly outperforms NLL on both answer relevance and overall quality without sacrificing grammatical correctness.

4. CONCLUSIONS AND FUTURE WORK

We have introduced NLL-QA, a simple yet effective criterion to replace the traditionally used NLL criterion in the post-processing step during the question generation process using neural question generation models. Through various evaluations, we have shown the promise of NLL-QA in improving neural question generation performance without changing or fine-tuning the question generation model itself.

The major challenge is defining a proper metric to evaluate the generated questions. NLL-QA is as reasonable criterion if we our metrics are grammatical correctness and relevancy to the input text. These two metrics are obviously unsatisfactory in educational domain because we also care about the “educational value” of a generated question, which is difficult to quantify, let alone doing so automatically. Therefore, I would like to discuss ways to define quantitative metrics for evaluating questions and methods to do so automatically.

5. REFERENCES

- [1] M. Agarwal, R. Shah, and P. Mannem. Automatic question generation using discourse cues. In *Proc. Workshop on*

⁴For binomial tests, questions in “Ties” category are split into their original preference categories that human raters selected.

Innovative Use of NLP for Building Educational Applications, pages 1–9, Jun. 2011.

- [2] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions. In *Proc. Annual Meeting of the Association for Computational Linguistics*, pages 1870–1879, Jul. 2017.
- [3] X. Du, J. Shao, and C. Cardie. Learning to ask: Neural question generation for reading comprehension. In *Proc. Annual Meeting of the Association for Computational Linguistics*, pages 1342–1352, Jul. 2017.
- [4] N. Duan, D. Tang, P. Chen, and M. Zhou. Question generation for question answering. In *Proc. Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Sept. 2017.
- [5] M. Heilman. *Automatic Factual Question Generation from Text*. PhD thesis, 2011.
- [6] A. Lavie and A. Agarwal. Meteor: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proc. Workshop on Statistical Machine Translation*, pages 228–231, Jun. 2007.
- [7] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Proc. Workshop on Text Summarization Branches Out*, pages 74–81, Jul. 2004.
- [8] M. Liu, R. A. Calvo, and V. Rus. Automatic question generation for literature review writing support. In *Proc. International Conference on Intelligent Tutoring Systems*, pages 45–54, Jun. 2010.
- [9] S. Miranda, G. R. Mangione, F. Orciuoli, M. Gaeta, and V. Loia. Automatic generation of assessment objects and remedial works for moocs. In *Proc. International Conference on Information Technology Based Higher Education and Training*, pages 1–8, Oct. 2013.
- [10] A. Papasalouros, K. Kanaris, and K. I. Kotis. Automatic generation of multiple choice questions from domain ontologies. In *Proc. International Conference on E-Learning*, pages 427–434, Jul. 2008.
- [11] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: A method for automatic evaluation of machine translation. In *Proc. Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Jul. 2002.
- [12] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proc. Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Nov. 2016.
- [13] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi. Bidirectional Attention Flow for Machine Comprehension. In *Proc. International Conference on Learning Representations*, Apr. 2017.
- [14] D. Tang, N. Duan, T. Qin, Z. Yan, and M. Zhou. Question Answering and Question Generation as Dual Tasks. *ArXiv e-prints*, abs/1706.02027, Jun. 2017.
- [15] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proc. Annual Meeting of the Association for Computational Linguistics*, pages 189–198, Jul. 2017.
- [16] Z. Wang, A. S. Lan, W. Nie, A. E. Waters, P. J. Grimaldi, and B. R. G. Qg-net: A data-driven question generation model for educational content. In *Proc. Annu. Conf. Learn. Scale*, pages 7:1–7:10, Jun. 2018.
- [17] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, ĀAukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean. Google’s neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.
- [18] Z. Yang, J. Hu, R. Salakhutdinov, and W. W. Cohen. Semi-supervised QA with generative domain-adaptive nets. In *Proc. Annual Meeting of the Association for Computational Linguistics*, pages 1040–1050, Jul. 2017.
- [19] X. Yuan, T. Wang, C. Gulcehre, A. Sordoni, P. Bachman, S. Subramanian, S. Zhang, and A. Trischler. Machine Comprehension by Text-to-Text Neural Question Generation. In *Proc. Workshop on Representation Learning for NLP*, pages 15–25, May 2017.
- [20] Q. Zhou, N. Yang, F. Wei, C. Tan, H. Bao, and M. Zhou. Neural Question Generation from Text: A Preliminary Study. In *Proc. National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671, Nov. 2017.

Beyond Autoscoring: Extracting Conceptual Connections from Essays for Classroom Instruction

Korah J. Wiley, Allison Bradford, Zachary Pardos, Marcia C. Linn
University of California, Berkeley
Berkeley, CA 94720-1670
{korah.wiley, allison_bradford, pardos, mclinn}@berkeley.edu

ABSTRACT

While automated essay evaluation techniques have dramatically reduced instructors' grading burden, they fall short of providing instructors with the rich qualitative insights into students' sense making process that a careful read of essays can afford. In this study, we demonstrate how word embedding techniques can serve as a complement to automated scoring, providing instructors with valuable near-time insight into how their students are conceptualizing targeted lesson concepts. For this study, we use a post-test essay associated with two Web-based Inquiry Science Environment (WISE) units that provides instruction about how the sun causes increases in temperature. We create word2vec models fit to students' c-rater scored essay responses at each score level of a rubric designed to assess students' integrated understanding of targeted concepts. Using cosine similarity, we identify, with statistically reliability, the ideas that students at each score level of the rubric used in relation to the concepts targeted in the essay prompt. Our instructor interview reveals the validity of the results in providing insight into students' ideas, differentiated across understanding levels.

1. INTRODUCTION

In the domain of science, new science education reform efforts, like the Next Generation Science Standards (NGSS), call for students to both coherently understand and communicate complex science ideas[6]. Consequently, essays that assess students' developing knowledge of complex science ideas could increase the validity of assessment in science classrooms[6]. Concomitant with the call for increased use of essay assessments is the need for machine-based techniques to quickly and reliably analyze student essays in order to provide instructors with qualitative insights about how their students are developing and connecting complex science concepts.

Advances in the field of natural language processing (NLP) have given rise to automated essay evaluation (AEE) tech-

niques that help instructors meet the challenges of essay scoring. However, there is still the need to develop effective techniques to assess and support students' comprehension of complex ideas expressed in their essays[9]. In order for instructors to provide targeted support based on their students' developing ideas, they need to have the qualitative insight that comes from reading the essays. Without it, instructors are left in the dark regarding the different ways their students make sense of the targeted concepts being assessed by the essay item. Even though instructors may have access to the exemplars used to train AEE models, the distance between the exemplar responses and that of their students can leave instructors guessing about the true nature and quality of their students' understanding. In this study, we describe the development and evaluation of word2vec models that augment the value of automated scoring by analyzing and comparing the conceptual connections that students in different scoring categories express in their essay.

2. BACKGROUND

Research in the field of teaching and learning has shown students' ability to develop an integrated understanding of complex ideas, such energy transfer and transformation, is influenced by how well their instructors notice and understand their ideas[8]. Therefore, to support students in communicating integrated understanding of complex ideas, it is important to develop machine-based analyses that provide teachers opportunities to see, understand, and respond to student ideas.

2.1 Auto-scored Student Essays

In partnership with Educational Testing Services (ETS), we have previously used the c-rater algorithm to score student essays from various assessment items in Web-based Inquiry Science Environment (WISE) units[2]. The student responses used to train the c-rater model were human-coded using a rubric based on the Knowledge Integration (KI) framework[4]. The KI framework supports students to sort through their ideas to develop an integrated understanding of normative science concepts[3]. Since the rubric used to score the essay assessment items in the units prioritizes the links that students make between normative science ideas, the c-rater generated scores reflect the extent to which students' ideas are normative and linked.

To support instruction based on students' ideas, we create skip-gram models[5] of students' c-rater-scored essay re-

Korah Wiley, Allison Bradford, Zach Pardos and Marcia Linn
"Beyond Autoscoring: Extracting Conceptual Connections from Essays for Classroom Instruction" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 718 - 721

sponses. In doing so, we were able to augment automated essay feedback and, thus, provide instructors with quantitative information regarding the correctness of students' ideas and qualitative information regarding the connectedness of their ideas.

2.2 Science Content Knowledge

The WISE units used in this study were designed to support students in developing an integrated understanding of how energy from the sun is transferred and transformed and how those processes cause an increase in temperature. To develop a normative and integrated understanding of this complex concept, students need to link the following ideas:

- Energy from the sun travels to Earth in the form of solar radiation.
- Solar radiation is in part reflected and absorbed by objects.
- Once absorbed, solar radiation is transformed into heat energy, resulting in an increase in temperature.
- Objects that absorb solar radiation can re-emit that energy in the form of infrared radiation.
- Infrared radiation is blocked and reflected by certain objects (e.g. atmospheric gasses, glass).
- Like solar radiation, infrared radiation is in part reflected and absorbed by objects.
- Once absorbed, infrared radiation is transformed into heat energy, resulting in an increase in temperature.

The assumption is that students at each score level conceptualize and connect these science ideas differently. Since the word2vec models encode the semantic meaning of the words as vectors, we can, for example, analyze the differentiated ways students scoring at the level of 2 conceptualize "temperature" as compared to how students at a level 5 conceptualize "temperature". Using cosine similarity, we can provide teachers with a list of the words that their students have conceptually connected to the target lesson concept (e.g. temperature).

Given this potential, we conducted this study to address the following research questions:

1. Can we develop a skip-gram model from pre-scored essay responses to reliably extract the differential ways students develop and conceptually connect complex science ideas?
2. Do the skip-gram models generate results that are pedagogically informative (i.e. support teachers to notice, understanding, and respond to their students ideas)?

3. METHODOLOGY

3.1 Offline Model Development (RQ1)

Student essay responses were prepared for skip-gram development using standard Python libraries, pywsd and NLTK were used to perform basic NLP techniques (i.e. case-adjustment,

abbreviation expansion, punctuation and symbol exclusion, lemmatization, and stopword modification and removal)[1]. Skip-gram models for student responses at the KI rubric score levels (range 2-5, four total) were developed using the gensim library[7]. Model hyperparameters were adjusted to produce cosine similarity results to the various target words that a content-expert validated as conceptually relevant to the prompt. To extract the meaning of a word as used and understood by the student rather than its conventional meaning, we created the word vectors from the study datasets rather than using pre-trained vectors.

3.2 Classroom Evaluation (RQ2)

To determine the efficacy and utility of our skip-gram models to support teacher noticing of student ideas, we conducted a semi-structured interview of the high-school physics instructor whose students' essay responses generated the model results referenced during the interview.

3.3 Datasets

The dataset used in the offline development and evaluation of the skip-gram models consisted of student text responses to the explanation portion of the following post-test essay assessment item:

Let's think about how global warming happens. On a COLD day, Akbar walks to his car that is parked in the sun and has not been driven for a week. Predict the temperature inside the car:

- Colder than the outside air
- Warmer than the outside air
- Exactly the same as the outside air

Explain your answer:

The students of four sixth grade science teachers from two Bay Area middle schools (N=497) generated their responses to this prompt after engaging the WISE Global Climate Change unit (<https://wise.berkeley.edu/project/24751>). In this unit, students learned how solar radiation from the sun can be absorbed, transformed into heat energy, and trapped by greenhouse gases as infrared radiation, which leads to increased temperature.

Student responses were, on average, 34 words long and were human-coded, from 1 (low) to 5 (high), using a KI rubric. Responses received a score of 1 if they were off-task or irrelevant and, thus, were excluded from the model development dataset. Responses that included normative but unconnected ideas about the transformation of solar radiation to heat received a score of 3. Responses that connected one or more normative ideas about the transformation of solar radiation to heat received a score of 4 or 5, respectively. The distribution of scores were as follows: Score 2 = 243; Score 3 = 92; Score 4 = 42; Score 5 = 23.

The dataset used in the classroom evaluation of the skip-gram models consisted of student text responses to the same essay prompt used for the offline model development and evaluation (see above). The students of a ninth grade physics

Table 1: The cosine similarity results for the target word "temperature" of the skip-gram models associated with each score level using the development dataset. The vocabulary size of each model is in parentheses.

Temperature							
Score = 2 (105)	cos_sim	Score = 3 (55)	cos_sim	Score = 4 (62)	cos_sim	Score = 5 (58)	cos_sim
become	0.999	get	0.998	inside	0.985	get	0.987
reason	0.999	into	0.994	get	0.985	glass	0.981
colder	0.999	become	0.987	glass	0.981	inside	0.981
inside	0.999	glass	0.986	into	0.969	become	0.976
time	0.996	inside	0.977	become	0.968	into	0.961
get	0.995	particle	0.967	travel	0.967	car	0.935
hot	0.995	may	0.933	radiation	0.955	bounce	0.906
warmer	0.993	bounce	0.932	bounce	0.922	pass	0.87

Table 2: The most similar words and associated p-values for the target word "heat" of the skip-gram models associated with each score level using the development dataset. The vocabulary size of each model is in parentheses.

Heat							
Score = 2 (105)	p-value	Score = 3 (55)	p-value	Score = 4 (62)	p-value	Score = 5 (58)	p-value
outside	<1.0E-6	outside	2.42E-04	car	1.13E-04	car	0.009
inside	<1.0E-6	inside	0.012	air	0.007	glass	0.009
temperature	<1.0E-6	air	0.049	glass	0.045	infrared	0.009
cold	<1.0E-6	energy	0.049	radiation	0.045	energy	0.042
air	<1.0E-6	get	0.049	turn	0.045	day	0.042
warm	<1.0E-6	into	0.049			light	0.042
get	3.58E-06	warm	0.049			solar	0.042
sun	0.039	trap	0.049				

instructor from a Bay Area high school generated their responses (N=155) before and after engaging the WISE Solar Ovens unit, which focused on designing, building, and testing a solar oven (<https://wise.berkeley.edu/project/24537>). In this unit, students learned about the same energy cycle described in the sixth grade unit, and then explored how different designs influence that energy cycle and temperature change in a solar oven.

Student responses were, on average, 30 words long and were scored using a c-rater algorithm based on the same KI rubric used for the model development dataset. Similarly, responses received a score of 1 if they were off-task or irrelevant and were excluded from the dataset. The distribution of scores were as follows: Score 2 = 82; Score 3 = 46; Score 4 = 22; Score 5 = 3. Due to sparseness of responses at the score level 5, a skip-gram model was not created for this score level.

4. RESULTS

4.1 Offline Model Development (RQ1)

We chose "heat" and "temperature" as target words for our model evaluation, since the essay used in this study was designed to assess students understanding of how solar radiation and infrared radiation cause an increase in temperature. Our initial cosine similarity results from the four skip-gram models appeared to be consistent with our content expert's expectations regarding the ways that students are different scoring levels would connect ideas related to "temperature" (see Table 1). However, these initial results were not reproducible from one model run to the next, in terms of exact words and relative cosine similarity rank. The top 8 results for each model within a given run displayed patterns con-

sistent with the content expert's expectations, albeit with variable reproducibility.

We used distribution probability to establish the statistical reliability of our model results. Specifically, we used hypergeometric and binomial distribution test, respectively, to determine the probability that any given word in the model's vocabulary would appear in the top 8 cosine similarity results and do so consistently enough to yield a p-value < 0.05. We ran each model 12 consecutive times on a random sample of the essay responses, where the number of responses in the sample equaled the total number of responses in the dataset. After 12 consecutive runs of each model, words would need to appear in the top 8 cosine similarity results of 3 or 4 of the runs, depending on the model's vocabulary size, to achieve a p-value < 0.05 according to the reliability test.

Using the development dataset, we generated a statistically reliable list of the words based on the model results for the target word "heat" (see Table 2). Examination of this list revealed conceptually meaningful differences across the scoring levels, as confirmed by the content expert and physics instructor. The most statistically reliable results using the target word "heat" were: score level 2 model: air, temperature, inside; score level 3 model: inside, energy, trap; score level 4 model: car, air, radiation; and score level 5 model: car, radiation, trap (see Table 2). These model results indicate a typical progression of student ideas when they are beginning to understand the mechanism of how solar radiation transforms to heat.

Table 3: The most similar words for the target word "temperature" of the skip-gram models associated with each score level using the classroom dataset. The vocabulary size of each model is in parentheses.

Temperature		
Score = 2 (92)	Score = 3 (68)	Score = 4 (62)
become	get	inside
reason	into	get
colder	become	glass
inside	glass	into
time	inside	become
get	particle	travel
hot	may	radiation
warmer	bounce	bounce

That the lists of words produced from the target word "temperature" are similar to those from the target word "heat" reflects the similar ways in which students conceptualize "temperature" and "heat".

4.2 Classroom Evaluation (RQ2)

To investigate the alignment of the model results with the instructor's expectations of the conceptual connections that students at each score level would make, we presented the instructor with the cosine similarity results from each model for the target word "temperature" (see Table 3). Although the statistical reliability for the model results had not yet been established, we asked the instructor to interpret the results for each individual model and cross-comparatively. The instructor commented that the results from the model resonated with her expectations of what a developed understanding looks like and what a still-developing understanding looks like.

Furthermore, we asked the instructor when and she might use the information provided by our models. She indicated that she would use a tool like this as a form of formative assessment to see what her students' ideas were while instruction was still ongoing. She went on to say that she could use the information from the models to inform her instruction, and that it would help her to decide if she needed to have a whole class conversation, do a demo, or show a video to support her students in sorting and integrating their ideas towards normative understandings.

5. SUMMARY AND ADVICE SOUGHT

Our ability to create meaningful embedding models of the differentiated way students at each scoring level conceptualize target ideas has several implications:

- A strategy for extracting statistically reliable and meaningful results from small, unevenly distributed datasets typical of K-12 classrooms and upper-division college courses
- The revelation of subpopulation-specific conceptualizations of target lesson concepts
- The realization of the full assessment power of essays in an online learning environment

For this doctoral consortium, I would like advice regarding how to further validate my models. Since my approach utilizes relatively small datasets to generate instructor/class-specific results, I would like to discuss additional ways to establish the reliability of my model results. Beyond distribution probability, what other statistical methods could I use to establish reliability? Which other approaches would serve as comparison cases for further model evaluation? Furthermore, to which other domains should we apply our model?

A major factor to consider in the consortium discussion is what instructors have identified as a barrier for use, namely model understandability. Instructors with whom we partner have stressed the importance of being able to understand how the models that automatically assess student responses work. Importantly, their confidence in the model results is linked to the extent to which they understand how the model generates the results. Therefore, it is a priority to develop a model that both generates meaningful results and can be easily explained.

Our results are promising and, through this consortium, we hope to determine how best to further develop our approach for essay evaluation so that both researchers and practitioners can be confident in the results.

6. REFERENCES

- [1] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed 07 September 2018].
- [2] C. Leacock and M. Chodorow. Crater: Automated Scoring of Short-Answer Questions. *Language Resources and Evaluation - LRE*, 37:389–405, 2003.
- [3] M. C. Linn and B.-S. Eylon. *Science learning and instruction: taking advantage of technology to promote knowledge integration*. Routledge, New York, 2011. OCLC: ocn176946367.
- [4] O. L. Liu, J. A. Rios, M. Heilman, L. Gerard, and M. C. Linn. Validation of automated scoring of science assessments. *Journal of Research in Science Teaching*, 53(2):215–233, Feb. 2016.
- [5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13*, pages 3111–3119, USA, 2013. Curran Associates Inc.
- [6] National Research Council. *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas*. The National Academies Press, Washington, DC, 2012.
- [7] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. <http://is.muni.cz/publication/884893/en>.
- [8] A. D. Robertson, R. E. Scherr, and D. Hammer, editors. *Responsive teaching in science and mathematics*. Teaching and learning in science series. Routledge, Taylor & Francis Group, New York, 2016.
- [9] K. Zupanc and Z. Bosnic. Advances in the field of automated essay evaluation. 39:383–395, 2015.

Predicting student academic outcomes in UK secondary phase education: an architecture for machine learning and user interaction

Matthew Woodruff
Department of Computer Science
University of Surrey
Guildford, Surrey, UK.

m.woodruff@surrey.ac.uk

ABSTRACT

Groups of schools face significant challenge to ensure that student assessment is consistent and valid, that predictions are accurate and that data policy does not adversely impact teacher workload. This study looked to determine if student outcomes at the end of secondary phase education in the UK, and by extension elsewhere in the world, could be reliably predicted. Data from two academic years, twenty-three secondary schools representing cohorts of approximately four thousand students were used to conduct comparative analysis on multiple machine learning experiments utilising Microsoft Azure Machine Learning and code written in Python. Findings indicate that Attainment 8, the main performance indicator used by the UK Department for Education, could be predicted by linear regression models and neural network approaches with an accuracy in line with human (teacher) predictions, and when combined with human predictions the models provided significant ($P=0.00039$) gains on either approach in isolation. Seeking advice on further research particularly to introduce interactions to include a 'human in the loop' on model inputs, to refine feature selection, and to predict curriculum level variations in order for human intervention to occur early and improve outcomes.

Keywords

Predictive analytics, Machine Learning, Student outcomes, Python, Azure, Bot, human in the loop.

1. INTRODUCTION

'Progress 8' is an important measure in the current accountability framework for secondary phase state funded education in the UK [1]. Many countries have similar national tests, such as the National Assessment of Educational Progress (NAEP) in the US or the NAPLAN tests in Australia [2]. In brief, it measures the progress a student makes in academic study across their best eight subjects. It measures this from their attainment at point of exit from primary phase education (aged 12 / National Curriculum Year 6), and its basis is relative to average attainment of the prior year's cohort (at age 16 / National Curriculum Year 11), from the same starting

point. Progress 8 is used both at a student level, and aggregated in average to provide a 'score' at an institution level. A score of 0 represents a student (or school) making the average progress for students of the same prior attainment in the prior year's cohort. A score of 1 would represent that the student has made on average one grade more progress in each of their subjects (and conversely, -1 meaning one grade less progress).

Progress 8, in and of itself, is of limited use for formative or summative assessment at a student or school level. The data from which it is derived would be more appropriate for these purposes, namely subject level concept understanding, application and progression. Despite the inherent difficulties in reliably predicting Progress 8 [3], it is common place for schools to do so. In part, this is due to its nature as a high-stakes key performance indicator – one upon which the quality of education provided by the school is judged. Education policy in recent years in the UK has driven schools to become part of Multi Academy Trusts (MATs), and 72% of secondary phase education is now delivered by schools with Academy status [4], and part of a MAT.

The study involved a large MAT with a geographical spread and accountable for schools in all phases in both the state and the independent sectors. In 2018 the MAT formed a project partnership with Microsoft and Coscole, a specialist company focusing on education data and analytics. In addressing the need to predict Progress 8 and therefore understand how and where to intervene to support its students, the MAT surmised:

- The methods adopted for teacher led subject predictions are variable across schools, and within them
- Historical variance between teacher predicted outcomes and actual outcomes have not been easy to explain
- Significant time can be spent by staff, leadership teams and trusts in the production and analysis of data relating to likely outcomes
- Patterns of human predictive accuracy were not consistent between academic years

Given these challenges two key questions were posed:

- Can machine learning methods achieve human like predictive capability (or better) for Progress 8?
- If so, could the administrative burden placed on schools in the production of these metrics be reduced?

Matthew Woodruff "Predicting student academic outcomes in UK secondary phase education: an architecture for machine learning and user interaction" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 722 - 725

2. METHODS

If the results of this study were favourable in terms of predictive accuracy of outcomes it would be important to the study participants that there would be a mechanism for the approach to be repeatable, and to scale. A solution architecture was therefore identified at the outset that would support both of these considerations. The study comprised of data from 23 secondary schools, with a population of over 4000 students in their final year of study (National Curriculum Year 11, age 16).

2.1 Solution components

Figure 1 shows the high-level design that was created to support the modelling process and a proof of concept (POC) environment.

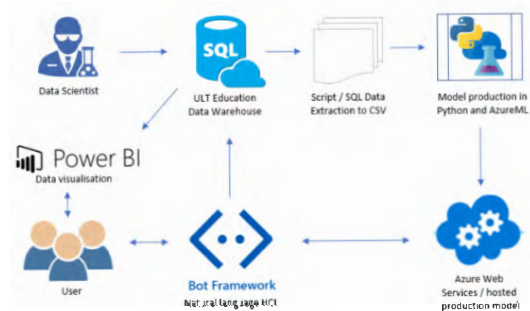


Figure 1. Proof of concept high level design.

2.1.1 Human computer interaction

The requirement for the human-computer interaction was to ensure that the supply of data, and the return of predictive output could be transacted by an education practitioner with little technical expertise. We identified existing channels in the MAT environment (and common in many MATs) for this interaction to occur and provided a ‘Chat Bot’ style interface in order to achieve the transfer of data, machine learning orchestration and for the supply of information back to the user. An overview of this process can be seen in Figure 2, below.

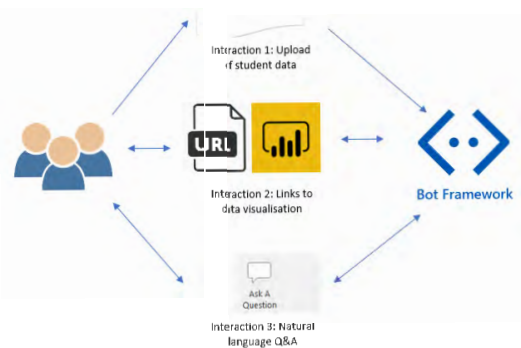


Figure 2. Human computer interaction POC scenario.

This interaction outputs predictive data to table outputs in Microsoft SQL Azure, which can be consumed by the Bot framework and/or by the data visualisation layer.

2.1.2 Data visualisation

In addition to the supply of predictive output back to the practitioner by way of the Bot interaction, in depth data visualisation was provided by Microsoft Power BI. When the machine learning predictive experiment web service completes, the resulting predictive data is posted to an Azure database table. Reports were created for data visualisation that consumed data from this table. The user could either navigate to the Power BI workspace where the report resides, or follow a link that that Bot posts to the chat window (should the session still be active). The link may include the refreshing of the report content, ensuring that the view of the data is always current.

3. Model features

3.1 Student characteristics

Common student characteristics exist that form the basis of much analysis completed in this sector. The annual government league tables, and other information that is available publicly, includes dimensions on gender, ethnic origin, prior attainment, ‘pupil premium’ where additional government funding is given due to disadvantage, and special educational needs. Given a body of research drawing correlations between these characteristics and educational attainment [5] they were included in the model features.

Student characteristics were converted into Boolean form (for example ‘Is Male’, ‘Is Special Educational Needs’, or ‘Is White British’)

3.2 English and Maths ‘Mock’ examination results

The MAT has a standard curriculum for ‘Mock’ examinations taken by Year 11 student in their final year of study. We could therefore use 2016/2017 exam data as a feature input as data on the same basis would be available for 2017/2018. This data is an integer scale (1-9), and also was likely to be correlated to the Progress 8 outcome (as Progress 8 double weights English and Maths). Other English and Maths data was available; however, it was excluded from our feature selection as it formed compound measures on the grade.

3.3 Prior attainment

Logic suggests that a student’s prior academic performance would be a good indicator of their future academic performance, a conclusion that is well supported by evidence [5]. Prior attainment data was available in the form of the student’s average attainment from the end of primary phase education (their ‘Key Stage 2 Average Fine Level’).

3.4 Teacher predictions

Data was available at student level with a teacher’s prediction of ‘likely’ grade at a subject level (for example in English Language, Mathematics, History and so on). Initially this feature, as a composite measure ‘Teacher Predicted Attainment 8’ was withheld from the machine learning experiment, and later introduced for comparative purposes.

4. Machine learning

4.1 Azure Machine Learning

Azure Machine Learning Studio was used to build several iterations of experiments. It was used to:

- Conduct feature analysis to understand which components of our data were most significantly correlated to the target (Attainment 8). This was performed by the Filter Based Feature Selection module using Pearson's correlation method, following the test/train split.
- Understand which forms of regression algorithm would be best suited to the type of predictive output
- Tune the resulting model hyperparameters for the smallest error

The final experiment utilised the Boosted Decision Tree Regression to create an ensemble of regression trees using boosting (where each tree is dependent on prior trees). This algorithm learns by fitting the residual of the trees that preceded it. The Azure ML implementation uses the Multiple Additive Regression Trees (MART) gradient decent boosting algorithm [6]. The filter-based feature selection module was used during the initial experiment. However, this was not necessary using the boosted decision tree algorithm due to its internally created feature summary (whereby features with zero weights are not used by any tree splits).

During the experiment the Azure ML function 'Tune Model Hyperparameters' was used and the results set to the recommendation of 25 leaves per tree, 25 samples per leaf node, a learning rate of 0.167076 and a total number of 125 trees were constructed.

4.2 Python

In addition to the Azure ML experiments a Python model was built as a comparative approach. This was completed not as a direct methods comparison, but as an illustration of the capability for Azure ML to host Python scripts within its data flow, and therefore to have complete control over all model features.

Initially, a Neural Network was used in the Python implementation and simplified to input (64 nodes), output and a single hidden layer (64 nodes). Its optimiser was set to use the RMS Prop Optimiser [7] and its learning rate set to 0.001.

When the validation error was no longer reducing over a defined number of epochs (20), we stopped the training.

Following this, a number of comparative algorithms were used which included a decision tree, a boosted decisions tree, a 2-layer neural network and a linear regression. In addition to the variation in algorithms, the experiments were modified to include or exclude 'Teacher prediction' data from the model, and also to extend the feature set to include further data related to the English and Maths results. The Python experiments were written in a VSCode environment using the Anaconda 3.0 deployment.

5. Results

It was realised early in the study, following unpredictable model outputs, that directly predicting Progress 8 was not going to be successful. It is beyond the scope of this paper to explain in depth the Progress 8 definition, but in summary it is a compound measure that utilises 'Attainment 8' (the eight highest grade values) and then applies a conversion based on the student's prior attainment and last year's cohort average attainment to result in a decimal number distributed around zero.

Instead, it was determined that Progress 8 may be predicted by predicting the Attainment 8 score, and then applying the known conversion method ([1], p27) to convert to the Progress 8 score.

The root mean squared errors (RMSE) of eight algorithmic approaches are compared to the RMSE of the teacher (Human)

prediction in Figure 3. Here we can see that both of the Neural Network approaches, as well as the Linear Regression surpass the accuracy of the human prediction. Attainment 8 is a decimal in the range 0-88 (8 qualifications on a 0-9 scale) and an RMSE of 1 would represent 1 grade difference in total between predicted and actual achievement.

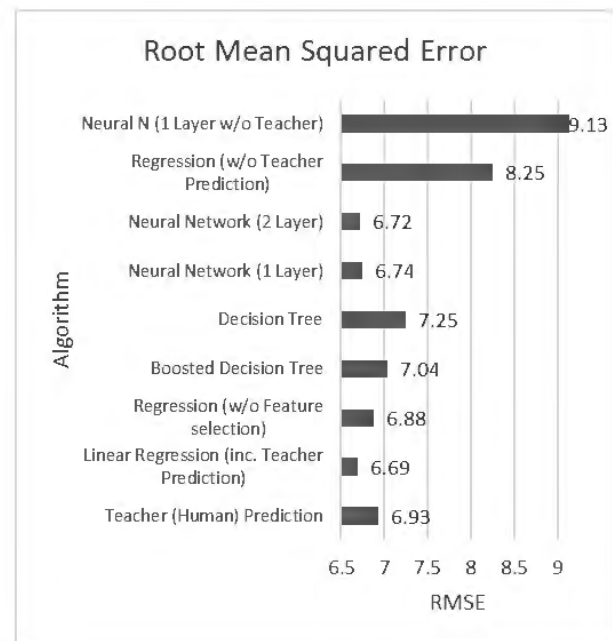


Figure 3. Root Mean Squared Errors of Python algorithms.

An F-Test was performed on the sample variance of the teacher prediction and the linear regression results to determine that the variances of the populations were unequal. A subsequent T-Test was conducted with the conclusion that the observed difference between the sample means is convincing enough to say that the average Attainment 8 result differs significantly ($p=0.000387$).

6. Discussion

This study set out to address two questions; Can machine learning methods achieve human like predictive capability (or better) for Progress 8, and if so, could the administrative burden placed on schools in the production of these metrics be reduced?

Initial findings suggest that machine learning approaches can produce output that is as reliable as human prediction. It could be argued that the time spent in collating such data, performing the analysis and communicating results can be reduced, therefore freeing time for education practitioners to focus their efforts in teaching young people and leading schools rather than wrangling data. However, in order to accomplish this at scale, system improvements would be required in terms of the automation of data collection, standardisation of assessment approaches, and in the human computer interaction to explain the meaning of results.

6.1 Human in the loop

We specified and delivered an architecture for data to be supplied by end users (defined by low technical, low data science knowledge), and the results to be understood by those users. We delivered this via a Bot Framework interaction for the upload of data and process trigger, and via data visualisation in Microsoft Power BI. We also included human input in the form of teacher predictions at a subject level.

We have not in this iteration, however, placed the human inside the feedback loop of the machine learning itself. There are opportunities to explore the following interactions. First, in data validation. Quality and clarity of data in educational settings, especially at an aggregated level, pose a significant challenge and our experiments eliminated data during the cleansing activity. Human input during this process may significantly improve the training data and help to reduce bias. The architecture could be extended to interact with the human to either input missing data, identify the correct rule-based approaches (for example where no prior attainment is available due to new entry into the UK state-based system) or to modify the feature weights (for example around the impact of disadvantage or ‘Pupil Premium’ at an individual level).

Second, what the human ‘knows’. Teacher interaction with students on a daily basis gives rise to a more complex understanding based on situational awareness and factors of data that are not easily captured formally. This might relate to wellbeing considerations, safeguarding and behaviours that are outside of the scope of data capture within the school.

Finally, although this is far from an exhaustive list, we could give more consideration to who the human is in this scenario. Obtaining data from students themselves, and on an ongoing basis, could trigger revisions to the predictive output.

6.2 Generalisability, bias, ‘explainability’, and ethics

The machine learning models in this study relied on a standard assessment practice in English and Maths mock exams, taken by all schools, with the same curriculum, at a similar time in the year, over two years (one for the training dataset and one for the test). It is not common for this structure to be in place, and so issues exist in how one might scale the predictive capability to other institutions and student populations. Many institutions utilise ‘Standardised Assessment’ practices in the Primary phase and Key stage 3 (age 11-14) using third party tests which to some extent addresses this issue, but these are rarely in place during Key Stage 4 (the focus of this study). There is a compelling reason to consider a longitudinal study on relative performance measures and other features that do remain consistent across schools.

For similar reasons we need to consider bias that may have been introduced in our model due to feature selection and the data cleansing activities. Notable examples exist where records may have been removed from training due to absent values, such as prior attainment. Prior attainment may be absent where a student has entered secondary state education either from private schools, from abroad, or in the case where parental consent was not given for the student to sit Primary exams. These characteristics of students might actually prove significantly correlated to outcomes, and bias may have been introduced in the way the training data was managed.

The algorithms used, and the HCI enabled is unable to provide the practitioner with an explanation of why a certain progress 8 score has been predicted at the level it has. This is important if machine output is to influence a decision on how public money might be spent to intervene to improve outcomes on its recommendation. It is recommended at a minimum that we include in the bot interaction or data visualisation confirmation of the weights we have applied to features in the regression, and have these communicated in a manner which can be understood. Further research is required to

understand how we might better describe, in human understandable terms, what path the machine has trod to reach its conclusions.

This leads fundamentally to our ethical considerations in predicting events that might lead a human to make a judgement on a student, and therefore tailor their educational experience in one direction or another. In an environment where all resources are scarce (notably funding) should a prediction be overstated, this might lead to a withdrawal of resources that would benefit the student and vice versa. The subjects must be aware of the manner in which data collected about them will be used, and it must be used in a way that supports the human decision-making process; it is not a replacement for it.

6.3 Impact to practitioners

We have identified, subject to certain challenges to be overcome with regard to standardisation and data collection, that a machine learning approach could provide additional predictive support and reduce workload. The next question is to what extent is this useful to practitioners to improve outcomes? The answer in this case is that it is at the very beginning of a series of activities that would prove more valuable for this purpose. As mastery of curriculum is broken down into subjects, the next level is to understand for those students that are predicted to miss their potential overall, which subjects in that composite measure are most at risk? Beyond P8 lies a cascading set of objectives to understand; which subjects are likely to have the furthest variance from expected standard? Which topics within those subjects? Which foundation principles need reinforcement?

7. REFERENCES

- [1] Department for Education. 2019. *Secondary Accountability Measures*. Available at <https://www.gov.uk/government/publications/progress-8-school-performance-measure>.
- [2] A Goss, P., Sonnemann, J., and Emslie, O. (2018). *Measuring student progress: A state-by-state report card*. Grattan Institute. Available at <https://grattan.edu.au/wp-content/uploads/2018/10/910-Mapping-Student-Progress.pdf>
- [3] Treadaway, M. 2016. *Don't try to forecast Progress 8!* Available at <https://ffteducationdatalab.org.uk/2016/02/dont-try-to-forecast-progress-8/>
- [4] National Audit Office. 2018. *Converting maintained schools to academies*. Available at <https://www.nao.org.uk/wp-content/uploads/2018/02/Converting-maintained-schools-to-academies-Summary.pdf>.
- [5] Sammons, P, Sylva, K, Melhuish, E, Siraj, I, Taggart, B, Toth K & Smees, R. 2014. *Influences on students' GCSE attainment and progress at age 16*. Available at https://dera.ioe.ac.uk/20875/1/RR352_-_Influences_on_Students_GCSE_Attainment_and_Progress_at_Age_16.pdf
- [6] Burges, C. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview* Available at <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf>
- [7] Hinton, G. 2016. *rmsprop: Divide the gradient by a running average of its recent magnitude*. Available at http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

Machine-Learned School Dropout Early Warning at Scale

S. Thomas Christie*
thomas.christie†

Daniel C. Jarratt*
daniel.jarratt†

Lukas A. Olson*
lukas.olson†

Taavi T. Taijala*
taavi.taijala†

ABSTRACT

Schools across the United States suffer from low on-time graduation rates. Targeted interventions help at-risk students meet graduation requirements in a timely manner, but identifying these students takes time and practice, as warning signs are often context-specific and reflected in a combination of attendance, social, and academic signals scattered across data sources. Extremely high caseloads for counselors compound the problem. At Infinite Campus, a large student information system provider, we modeled statistical relationships between student educational records and enrollment outcomes, using de-identified records and in-system analysis to guarantee student data privacy. The resulting risk scores are highly predictive, context-sensitive, nationally available, integrated into the existing student information system, and updated daily.

1. INTRODUCTION

Approximately 15% of American students do not graduate high school on time [16]. States and districts frequently employ interventions designed to improve educational outcomes, including reducing dropout rates. A key role of school counselors is to direct the application of these interventions to the students who need them most. Counselors first need to identify these students but are faced with information overload. Each student's data is distributed across a student information system (SIS) and often other systems or people, making it difficult to synthesize into an accurate, comprehensive portrait of a student's risk. Compounding the problem are extremely large counselor caseloads—the national average is 430 students per counselor—with higher numbers typical in schools serving children with other structural disadvantages [11].

Early warning systems function as “automated attention” for overworked counselors by automatically identifying students

who might benefit from additional institutional resources. They automate the more tedious data analysis and summarization tasks so that counselors can focus on what humans do best: building relationships.

1.1 Alternate approaches

An effective dropout prevention system requires developing people, processes, and technology [7], identifying valid predictors, managing data and reports, assigning interventions, and monitoring student progress [10, 15]. In this paper, we focus just on the technology that identifies risk factors and estimates student dropout risk, which can then be embedded in a larger dropout prevention system.

Quantitative and qualitative determination of school dropout risk factors is a decades-old area of research [19], though the mid-2000s were a particularly important inflection point. High-profile studies of dropouts in the Chicago [3] and Philadelphia [17] urban districts led to the development of statistical methods for determining risk factors and their incorporation into early warning systems. Several organizations, often working together, have been instrumental in encouraging American schools to adopt research-based best practices [10], including the U.S. Department of Education's Regional Education Laboratories [18], the Consortium on School Research [2] and NORC [7] at the University of Chicago, the American Institutes for Research [8], and the Everyone Graduates Center at Johns Hopkins University [9]. Most states now make an early warning system available to their school districts [6].

Until the mid-2010s, all widely-used dropout early warning systems used threshold-based models, characterized by a few easily comprehensible predictors with associated risk thresholds (e.g., failing at least one course or being absent at least 20 days). The simplicity and auditability of these models, and the associated ease of implementation using common software and spreadsheet skills, is their key advantage over machine-learned systems. Students are measured on each predictor and flagged as “on-track” or “off-track” based on which side of a preset threshold their data point falls. Staff can intervene with students who have the most “off-track” risk flags, or whose risk areas correlate with particular intervention domains. A particularly influential approach is Balfanz's “ABC” taxonomy, in which students are measured on attendance, behavior, and course performance metrics [4], optionally with different thresholds for different student subpopulations [12]. While some educational institutions

*Infinite Campus, Blaine, Minnesota, USA

†@infinitecampus.com

implemented an ABC-style system themselves [10], others used spreadsheets or data tools made available by organizations like American Institutes for Research [8]. An active area of research involves determining which predictors and thresholds are appropriate for each school, or whether single thresholds are appropriate at all.

To overcome the limitations inherent in threshold-based systems, we and other organizations created machine-learned dropout risk identification systems in the mid-2010s. A number of researchers describe their systems in the academic literature (e.g., [1]) or industry white papers (e.g., [20]). Several organizations serve machine-learned dropout predictions at the scale of hundreds of thousands of students in many school districts. The Wisconsin Department of Public Instruction’s Dropout Early Warning System uses data reported to the state every few months by the districts’ SISs to build machine-learned models. The system produces two predictions per year and is available for students in grades 6–9 [14]. Mazin Education (through BrightBytes) and Hoonuit both sell machine-learned early warning and intervention monitoring systems for all grade levels.

Machine-learned systems have two major advantages over threshold-based models. First, the additional model complexity affords more accurate predictions and allows system designers to infer which risk factors are predictive in the presence of other factors or for different populations. Second, the variety of model architectures allows for more than just inferring overall risk. Designers can choose, for instance, to model time until dropout so that staff can intervene according to acuteness of risk, or to model uncertainty.

1.2 Our contribution

Two key obstacles prevent machine-learned early warning systems from being deployed nationwide. First, the predictive quality of these models is chiefly a function of data availability, as models must be trained on a large dataset—including a variety of educational contexts and outcomes—to ensure they perform well for students they haven’t seen before. Models built on a single district’s or even state’s data may not generalize well to other populations. However, building a model on multiple states’ data requires the data to be standardized, and without a common SIS to enforce uniformity, manipulating data into a common format is costly and time consuming.

Second, predictions must be surfaced to educators in a frequent and easy-to-use manner, which means the most successful systems will be closest to existing daily workflows. In some existing systems, staff members must log into a separate software program to access risk scores. In others, there are months between score updates. Timely prediction is important; the sooner a school is aware of a student at risk, the more time it has to intervene.

Infinite Campus provides a large student information system and has made significant investment in education data localization and standardization, reporting and warehousing, and user workflows for American K–12 education. Our role in the industry positions us to address the key remaining gaps in early warning systems using centralized data warehouses, standardized data, and placement of the early warning ap-

plication into the existing SIS.

Contributions

Available nationally in 32 states yet contextual to each child’s educational environment

Useful predictions: highly predictive, daily updates, four risk category scores, with consistent of predictive quality between protected student groups

Integrated into the student information system with no imports, exports or synchronization necessary

In the following section we describe our implementation of a dropout early warning system on more than 6 million student-years of educational records across 32 states. Our overall dropout risk score has excellent predictive quality with an AUC of 0.941 (see table 1 for additional quality metrics), and includes additional machine-learned scores that help counselors understand the source of a student’s risk to guide which interventions may be appropriate. Risk scores, delivered automatically and updated daily, are integrated into our existing SIS and available to counselors as an enhancement to their existing workflows.

2. IMPLEMENTATION

System design. Student data is stored in a number of relational SQL Server databases for school staff to create, read, update, and delete educational records. These databases exist in Infinite Campus’s fully owned and operated Tier 4 data centers that fulfill security requirements of the U.S. Department of the Interior. Student records exist in a variety of data structures and are recorded with varied time granularity for 45 states and the U.S. federal government, a portion of which we use for early warning. Because our model architecture requires a common data structure, we aggregate student records into a fixed format with one row per student-year, where ‘year’ corresponds to an academic year. Aggregated data is periodically transferred to a central repository in the same data center. Data for past students whose educational outcomes are known (e.g., graduation or dropout) is then used to build a machine-learned model relating summarized educational records to student enrollment outcomes. The model, along with summarized properties of each district, school, and geographic area present in the dataset, is deployed behind an API. Each day, records for currently enrolled students are aggregated, de-identified, and sent to the API, which returns risk scores for each student. Figure 1 illustrates this architecture. The returned risk scores and a score history is made available to counselors via the SIS user interface. By integrating and automating the process of generating and updating risk scores into the SIS, we relieve dropout prevention teams from the burden of collecting, storing, and analyzing the source data themselves.

What we predict. Each student-year of aggregated educational records is tagged with one of three labels: “needs early warning”, “does not need early warning”, or “ignore for early warning”. A student-year is labeled with “needs early warning” if the student’s records include known undesirable outcomes during the year in question or future years. For example, records for an 8th grader in 2014 would be labeled

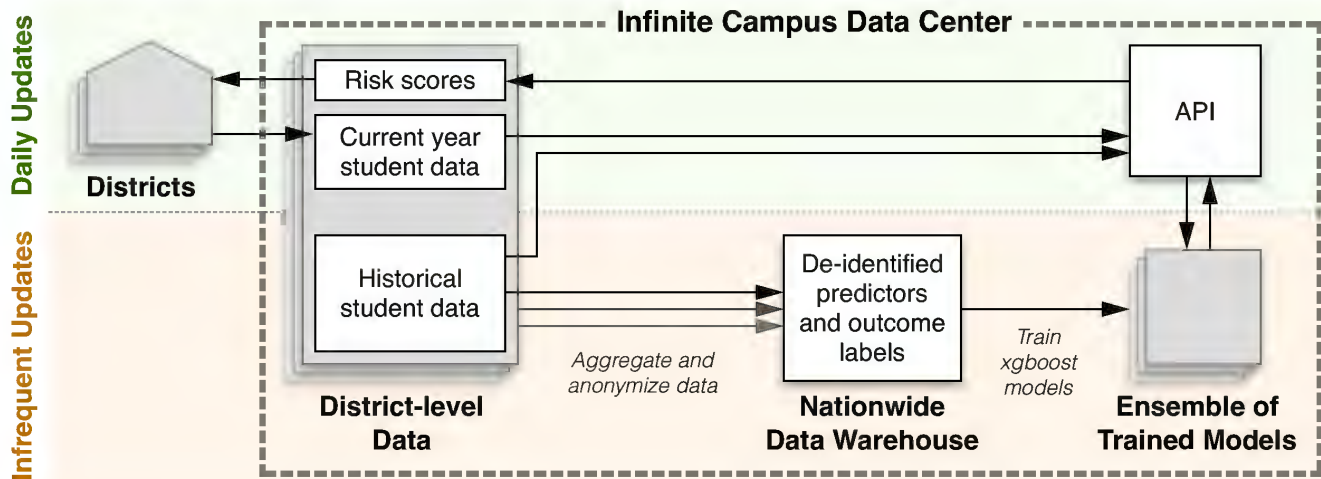


Figure 1: Schematic of architecture describing data flow for model training and prediction. Model training is performed infrequently, while risk scores are recalculated daily to incorporate new information. All data remains in the Infinite Campus data center and is not transferred to third-party servers. Identifying information is removed before transfer between systems.

“needs early warning” if they dropped out in 2016. We define undesirable outcomes based on enrollment end status codes identified by states as indicating school dropout, and expand the definition to include unsatisfactory academic progress (i.e., retention or demotion) and expulsion or other removal. A student-year is labeled with “does not need early warning” if the student had no undesirable enrollment outcomes during the year in question or future years, *and* we can confirm that the student persisted to graduation. If we do not know whether the student persisted to graduation—the student transferred outside of our system or is still enrolled—then the student’s data is censored and we lack ground truth labels for it. Each of these students’ years are labeled with “ignore for early warning” and excluded from training and evaluation. To translate state-specific enrollment status codes to outcome categories, codes were mapped by three independent raters, then differences were reconciled and validated by comparing resulting outcome rates across districts. Data from a school district are removed from training if that district has abnormally high “needs early warning” or “ignore for early warning” rates, as these conditions may indicate underlying inconsistencies in record-keeping that warrant further investigation. In addition, student-years are removed from training if the student’s cohort is not scheduled to have graduated yet, in order to remove label bias in earlier grades.

Our data collection and labeling process produces approximately 6.4 million rows of labeled data. We use 45% of rows for model training, 5% for model validation (to determine training stopping points), and 50% for final quality evaluation, split by student. Roughly 16% of training rows are labeled as “needs early warning”.

Predictors. Our training set is produced by collecting and summarizing educational information from the core SIS database. This summarized information relates to attendance, academic performance, behavior, household and enrollment

stability, and other items. We chose predictors that are consistent across states and districts and that are supported by the dropout prevention literature (e.g., [19]). Where data is localized, we employ experts who communicate directly with stakeholders in districts and states to ensure we understand the unique characteristics of local use and law. Attendance information includes the proportion of class time a student was actually present, as well as absences grouped by type of excuse. Academic performance information includes the proportion of course grades attributed to each letter grade, overall high school GPA, and the proportion of attempted credits successfully earned. Behavior information includes the number of behavior infractions and resolutions, as well as whether weapons, drugs, or harassment were involved. Household and enrollment stability information includes the presence of past undesirable enrollment outcomes and how often the student changes home addresses, schools, or districts in the middle of school years. Finally, we include contextual information such as age and grade level. In total, we have approximately 70 distinct predictors per year, and each student-year row includes the current and previous year’s data.

One core design goal for our system was to have a nationwide statistical model that is sensitive to local and contextual factors. We achieved context-specific performance with two types of feature engineering: including subpopulation aggregation features and calculating interactions among a student’s personal features as well as the subpopulation aggregates that apply to them. For example, a student’s attendance or academic data *relative to their peers* in a given group may carry information about risk. To allow for this possibility, we calculated two types of summary statistics for each school, district, and ZIP code in our dataset. For numeric predictors, we calculated the mean value per group. For categorical predictors, we calculated the proportion-per-group of each category. These group-level contextual features allow us to capture signals about students’ environ-

ments that are more informative than simple group-membership variables. The group-level information about each student-year's specific district, school, and ZIP code is joined on each student-year row for model building and prediction.

By including a wide range of features characterizing individual students and their educational environments, we allow the machine learning model to determine the significance of each feature and the relationship between features as they relate to risk. The wide range of potentially predictive features is simply not available to most non-SIS vendors. By using a relatively complex feature set and modeling architecture, we are able to capture complex contextual relationships between students and their environments.

Explanatory scores. While machine learning affords high predictive accuracy and the ability to capture complex relationships between predictive features, this comes at the cost of reduced model interpretability. In early conversations with customers, we were frequently asked *why* a student received a given risk score. To answer this question, we supplement our overall risk score with two types of explanatory 'category' scores that provide insight into which parts of a student's record are contributing to their overall risk.

The first set of explanatory scores is based on Balfanz's "ABC" categories [10]—attendance, behavior, and course performance—and an additional "stability" score including measures of household and enrollment stability. Each of the four scores is produced by a separate model trained only on predictors from its respective category. By partitioning the predictors according to category, we in turn are able to disentangle the impact each category has on the overall risk score. The predictive quality of category scores is necessarily lower than the overall risk score, because the category-specific models use a strict subset of the overall model's predictors. However, these scores indicate whether each category of a student's predictors, when taken by itself, is characteristic of a student with undesirable future enrollment outcomes.

In addition to scores built on subsets of predictors from each category, we also build scores for each category using a "counterfactual" approach. That is, if a student's records improved in a certain area (but the rest of the student's records stay the same), how would their risk change? To answer this question, we replace the values for the "actionable" predictors in a category with values corresponding to exemplary performance. The resulting data represents an attainable ideal for each student—if he or she attended every class, earned perfect scores on every assignment, or never behaved inappropriately. This data is used to produce a counterfactual risk score for each of the four categories, which when subtracted from the student's actual overall risk score indicates the potential "room for improvement" in each category; these are our final four explanatory scores.

While picking values corresponding to exemplary performance appears intuitive (e.g., 4.0 GPA, 100% attendance), using them to artificially modify student data has the potential to push the resulting data points outside the space of training examples, leading to unpredictable model behav-

ior. Preliminary analysis found this to be a problem for some "obvious" exemplary values, leading us to select values experimentally instead. For each feature, we used a statistical model to find the optimal bin (range of values) corresponding to the lowest predicted risk, which we subsequently validated by checking that the proportion of actual dropouts was lowest for this bin. We chose a reasonable value from each optimal bin to represent exemplary performance.

2.1 Modeling technique

The system described here must operate at scale within an industry setting and be robust to messy and missing data. To achieve this, we use the **xgboost** package [5] for modeling, which constructs a series of simple decision trees. Unlike logistic regression or neural networks, **xgboost** is robust to the presence of outliers and appropriately handles missing values. The decision-tree structure of model components provides an integrated way to capture contextual relationships between individual predictors and group-level aggregates. **xgboost** supports parallelization of model training, so training scales well on enterprise server hardware. We use the 'binary:logistic' training objective, so that **xgboost** models produce the probability of a student-year being labeled as "needs early warning". We use the area under the receiver operating characteristic curve (AUC) as the **xgboost** evaluation metric. AUC measures the quality of sorting produced by the model, with a high value for AUC indicating that the model is correctly assigning higher probabilities to student-years labeled as "needs early warning" than to those labeled as "does not need early warning".

Our modeling strategy ensures robustness to noise in two ways. First, we heavily regularize our **xgboost** models by using a small tree depth and relatively few training rounds, reducing over-fitting and making the model more robust to small changes in student data (both during training and during prediction). Regularization makes it possible to provide high-quality predictions for unseen data, such as a student in the evaluation dataset or a new customer whose data was not included in model training. Second, for each score type we train an ensemble of 11 to 25 **xgboost** models, and use the median prediction of all models. This technique further reduces variability between model deployments.

Predictions. Predictions are refreshed daily by aggregating educational records of currently enrolled students and sending those aggregates to our API, as illustrated in Figure 1, which provides GRAD scores and category scores back to the SIS. This technique allows us to provide score updates more frequently than competitors, and eliminates the requirement for school districts to transfer or analyze data on their own. The SIS then displays risk scores to staff members that have been given access to the early warning tool by district administrators.

As described above, we train the model using aggregations from past entire student-years. However, daily predictions are made for currently enrolled students, whose current year records contain only a partial year of data. We used several methods to mitigate the mismatch with our training dataset. First, in addition to aggregates summarizing a student's data from the current academic year, we also in-

clude aggregates from the student’s previous year as predictors. This allows the model to observe student data for an already-completed year and it affords analysis of year-over-year changes. Second, some data is in the form of event counts that accumulate throughout the year, such as the number of missed periods or the number of behavior resolutions. This data is converted into a rate, such as missed periods per instructional day. Rates are then directly comparable at all points in the academic year. Since rate-converted values are sensitive to small changes at the beginning of the school year, we instead use an ‘estimated rate’ calculated as a weighted combination of the previous and current year’s rates. After about four weeks of school, the previous and current year’s rates are equally weighted, with the current year’s rate weighted more heavily after each additional day.

We convert the probability output from `xgboost` into a user-facing “GRAD score” that ranges from 50 to 150, where 50 indicates high likelihood of undesirable enrollment outcomes in the future and 150 indicates high likelihood of persistence to graduation. We also considered that counselors may have a greater need to distinguish between students with low and moderate risk rather than between students with high and very high risk. That is, a 0.1 change in dropout probability from 0.05 to 0.15 is more important than a change in probability from 0.75 to 0.85. We therefore transform the raw probabilities to ‘spread apart’ students at the low end of the probability range (low risk), while compressing probabilities at the high end of the range (high risk).

2.2 Model evaluation

We evaluated our overall and subscore models on an evaluation set containing approximately 3.2M student-years (50% of the total dataset) that were not used for model building or validation during training. Results are listed in Table 1 and represent, as far as we are aware, the highest predictive quality in the industry. In Table 2, we also list evaluation results for our overall model by protected subpopulation [13]: sex, race/ethnicity, grade level, and free/reduced meal eligibility (a proxy for socioeconomic status).

Because counselors do not see predicted probabilities, but rather ordered GRAD scores, we chose the area under the receiver operating characteristic curve (AUC) metric that measures whether students’ predictions are ordered in terms of actual risk. The AUC effective range is 0.0 (perfectly inversely sorted) to 1.0 (perfectly sorted), with 0.5 indicating random predictions.

Futhermore, because counselors will give additional institutional resources to the few percent of students predicted most at-risk, we chose precision and recall metrics that measure whether how well that most at-risk prediction category actually contains at-risk students. We evaluated precision at 10% (P@10) and recall at 10% (R@10) following the literature [1]. A key limitation of precision@*k* and recall@*k* occurs when *k* is less than the population’s condition-positive (“needs early warning”) rate of the training set is 0.158. For subpopulation evaluation, precision and recall at baseline (PR@b) is based on that subpopulation’s own condition-

Model	AUC	P@10	R@10	PR@16
GRAD Score	0.941	0.865	0.549	0.719
Academics	0.914	0.825	0.524	0.682
Attendance	0.852	0.648	0.411	0.547
Behavior	0.808	0.582	0.368	0.493
Stability	0.860	0.654	0.415	0.548

Table 1: Risk score quality evaluation

Subpopulation	+ rate	AUC	P@10	R@10	PR@b
Female	0.130	0.935	0.770	0.597	0.683
Male	0.185	0.941	0.921	0.498	0.742
Hispanic	0.207	0.925	0.923	0.449	0.725
Asian	0.065	0.927	0.470*	0.712*	0.620
AIAN	0.275	0.927	0.972	0.354	0.771
NHPI	0.096	0.935	0.605*	0.676*	0.641
2+ races	0.185	0.935	0.909	0.487	0.717
White	0.134	0.937	0.790	0.592	0.692
Black	0.268	0.940	0.986	0.371	0.783
Not stated	0.136	0.951	0.871	0.639	0.762
6 th grade	0.218	0.896	0.918	0.421	0.695
7 th grade	0.212	0.910	0.924	0.434	0.709
8 th grade	0.207	0.921	0.921	0.444	0.716
9 th grade	0.232	0.937	0.978	0.423	0.779
10 th grade	0.169	0.937	0.888	0.529	0.727
11 th grade	0.112	0.940	0.730	0.654	0.692
12 th grade	0.079	0.953	0.573*	0.728*	0.649
NSLP: Free	0.252	0.919	0.961	0.385	0.737
NSLP: Reduced	0.130	0.920	0.745	0.579	0.652
NSLP: Paid	0.097	0.942	0.683*	0.700*	0.690
NSLP: N/A	0.125	0.942	0.788	0.634	0.711

Table 2: Overall model risk score quality for subpopulations. ‘+ rate’ refers to the baseline ‘condition-positive’ negative enrollment outcome rate for that subpopulation’s current or future enrollments. NSLP refers to the National School Lunch Program. AIAN refers to American Indian and Alaska Native. NHPI refers to Native Hawaiian and Pacific Islander. * means that the ‘+ rate’ value is less than 0.1 and therefore the effective maximum range of P@10 and R@10 is less than 1.0 for that subpopulation.

positive (“needs early warning”) rate. The effective range of PR@16 and PR@b is 0.0 (completely incorrect) to 1.0 (completely correct), with the random prediction rate equivalent to the baseline rate for that population.

3. LIMITATIONS AND EXTENSIONS

The use of year-level aggregates in model training erases temporal relationships between individual event records, making the system relatively blind to *patterns* of individual events within an academic year. To address this limitation, we are exploring alternate modeling strategies capable of ingesting event-based data streams that are both more granular and of non-uniform length. A second limitation is our model’s focus on grades 6–12. Interventions are most successful when they are applied early [21]. The ability to provide meaningful risk indicators for younger students could significantly improve outcomes by helping counselors target interventions toward the students who need them most, at the point they can ben-

efit from them most. To do this, we must overcome several data-consistency related obstacles, the most pressing being a lack of long-term datasets that are consistent in the type of information collected over time and the quality/reliability of the collection.

Finally, although target labels were created using inter-rater reliability methods, research on state policies, and student outcome data, the labels have not been verified by representatives from each school district who could personally attest to the accuracy of a given student's outcome. We intend to make the target calculation available to schools and to implement a system for users to provide feedback on our product's predictive accuracy to allow us to verify our labels and to continue to improve the quality of predictions.

4. CONCLUSION

We built a decision support system that provides high-quality, context-sensitive risk predictions and is integrated into an SIS that thousands of counselors already use in their workflows. In doing so, we offer daily risk assessments to millions of currently enrolled middle and high school students across the country. By automatically identifying students who may benefit from additional institutional resources in the service of timely graduation, we fulfill a key component of the dropout prevention process.

5. ACKNOWLEDGMENTS

We gratefully acknowledge all teams at Infinite Campus who have supported our work. We offer a special thanks to the Kentucky Department of Education and Kentucky educators for early partnership and feedback in this project.

6. REFERENCES

- [1] E. Aguiar, H. Lakkaraju, N. Bhanpuri, D. Miller, B. Yuhas, and K. L. Addison. Who, when, and why: A machine learning approach to prioritizing students at risk of not graduating high school on time. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 93–102. ACM, 2015.
- [2] E. Allensworth. The use of ninth-grade early warning indicators to improve Chicago schools. *Journal of Education for Students Placed at Risk (JESPAR)*, 18(1):68–83, 2013.
- [3] E. M. Allensworth and J. Q. Easton. What matters for staying on-track and graduating in Chicago public high schools: A close look at course grades, failures, and attendance in the freshman year. Research report. <https://eric.ed.gov/?id=ED498350>, 2007.
- [4] R. Balfanz, J. H. Fox, J. M. Bridgeland, and M. McNaught. Grad nation: A guidebook to help communities tackle the dropout crisis. <https://files.eric.ed.gov/fulltext/ED505363.pdf>, 2009.
- [5] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.
- [6] College & Career Readiness & Success Center at American Institutes for Research. State profile comparison. <https://ccrscenter.org/ccrs-landscape/state-profile/new-state-profile-comparison>.
- [7] L. Cordeiro, R. Blank, L. Hansen, D. Leeds, and L. Selfa. Considering the best early warning system (EWS) model to fit your needs, May 2016.
- [8] A.-M. Faria, N. Sorensen, J. Heppen, J. Bowdon, S. Taylor, R. Eisner, and S. Foster. Getting students on track for graduation: Impacts of the early warning intervention and monitoring system after one year. <https://eric.ed.gov/?id=ED573814>, 2017.
- [9] J. H. Fox, E. S. Ingram, and J. L. Depaoli. For all kids: How Kentucky is closing the high school graduation gap for low-income students. <https://eric.ed.gov/?id=ED572766>, 2016.
- [10] S. Frazelle and A. Nagel. A practitioner's guide to implementing early warning systems. *Regional Educational Laboratory at Education Northwest*, 2015.
- [11] D. J. Gagnon and M. J. Mattingly. Most US school districts have low access to school counselors: Poor, diverse, and city school districts exhibit particularly high student-to-counselor ratios. <https://scholars.unh.edu/cgi/viewcontent.cgi?article=1285&context=carsey>, 2016.
- [12] Key data for an “early warning system” with on- and off-track indicators that become the basis for tiered interventions. <http://guidebook.americaspromise.org/tools-directory>.
- [13] M. Kearns, S. Neel, A. Roth, and Z. S. Wu. An empirical study of rich subgroup fairness for machine learning. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, pages 100–109, New York, NY, USA, 2019. ACM.
- [14] J. E. Knowles. Of Needles and Haystacks: Building an accurate statewide dropout early warning system in Wisconsin. *JEDM | Journal of Educational Data Mining*, 7(3):18–67, July 2015.
- [15] Minnesota Dept. of Education. MEIRS 2.0 Minnesota early indicator and response system guide, July 2018.
- [16] National Center for Education Statistics. High school graduation rates. <https://nces.ed.gov/fastfacts/display.asp?id=805>.
- [17] R. C. Neild and R. Balfanz. Unfulfilled promise: The dimensions and characteristics of Philadelphia's dropout crisis, 2000-2005. <https://eric.ed.gov/?id=ED538341>, 2006.
- [18] I. R. E. L. Program. Early warning systems. <https://ies.ed.gov/ncee/edlabs/projects/ews.asp>.
- [19] R. W. Rumberger and S. Lim. Why students drop out of school: A review of 25 years of research, Oct 2008.
- [20] M. Technet. ML predicts school dropout risk & boosts graduation rates, June 2015.
- [21] J. A. Temple, A. J. Reynolds, and W. T. Miedel. Can early intervention prevent high school dropout? evidence from the Chicago child-parent centers. *Urban Education*, 35(1):31–56, 2000.

A Better Cold-Start for Early Prediction of Student At-Risk Status in New School Districts

Chad Coleman
Teachers College, Columbia
University
525 W 120th St
New York, NY 10027
+1 347-205-4919
cjc2238@tc.columbia.edu

Ryan S Baker, Ph.D.
University of Pennsylvania
3700 Walnut St
Philadelphia, PA 19104
+1 215-573-2990
rybaker@upenn.edu

Shonte Stephenson, Ph.D.
BrightBytes
717 Market St STE 300
San Francisco, CA 94103
+1 877-433-4036
shonte@brightbytes.net

ABSTRACT

Determining which students are at risk of poorer outcomes -- such as dropping out, failing classes, or decreasing standardized examination scores -- has become an important area of research and practice in both K-12 and higher education. The detectors produced from this type of predictive modeling research are increasingly used in early warning systems to identify which students are at risk and intervene to support better outcomes. In K-12, it has become common practice to re-build and validate these detectors, district-by-district, due to different data semantics and risk factors for students in different districts. As these detectors become more widely used, however, it becomes desirable to also apply detectors in school districts without sufficient historical data to build a district-specific model. Novel approaches that can address the complex data challenges a new district presents are critical for extending the benefits of early warning systems to all students. Using an ensemble-based algorithm, we evaluate a model averaging approach that can generate a useful model for previously-unseen districts. During the ensembling process, our approach builds models for districts that have a significant amount of historical records and integrates them through averaging. We then use these models to generate predictions for districts suffering from high data missingness. Using this approach, we are able to predict student-at-risk status effectively for unseen districts, across a range of grade ranges, and achieve prediction goodness comparable to previously published models predicting at-risk

Keywords

Learning Analytics, High School Graduation, Machine Learning, Ensembling.

1. INTRODUCTION

Graduating from high school is an educational achievement that is strongly linked to gainful well-paying employment, higher personal income, better personal health, reduced risk of incarceration, and lowered reliance on social welfare programs [23, 2]. Fortunately, graduation rates have been rising in the United States, trending towards reaching 90% nationwide by the year 2020 [18]. While this is a positive accomplishment, it leaves millions of students not completing high school,

representing a continuing crisis within the American educational system. This crisis is not evenly distributed; in the USA, there are much higher dropout rates for African American, Native American, and Hispanic/Latinx students [35, 19], up to 4 times the rate for White students, as well as for learners from low-income families and with disabilities [37].

There is a clear benefit in completing a high school education, so why do we still see alarmingly high levels of high school dropout? A great deal of research has been conducted trying to answer this one question, with the hope that once identification is achieved educators and administrators can apply a preventative or remedial intervention to curb student dropout [11]. However, many factors appear to lead to student dropout, including lack of social support from parents, poor motivation, low self-esteem, parental educational achievement and value, and economic factors, making it difficult to create a single intervention that works for all students [19, 30]. While demographic factors correlate with eventual dropout, these indicators are not considered actionable. A school district generally does not have the capacity to improve a student's economic condition, nor is it possible to alter a student's racial identity or gender. As such, the educational research community has focused on more actionable factors such as behavior, attendance, engagement, and social-emotional learning [21]. The most successful interventions have attempted to address issues related to specific indicators while also attempting to improve overall student academic engagement [14]. There are a range of potential interventions and many are costly, driving a need to identify the students that could benefit most from specific forms of support. Identifying these students can be a difficult task [10] which has led to an ongoing effort within the educational research community to determine which students are at risk of not graduating from high school [20] to apply proactive interventions that can help get students back on track [8].

This goal, along with the growing availability of student data, has led to Early Warning Systems, which leverage statistical methods applied to historical student data in order to predict outcomes for new students. Early work on predicting high school graduation tended to use statistical methods in order to infer the relationship between graduation and indicators such as grades and attendance. For example, the seminal Chicago Model developed an "on-task" indicator built from first-year high school student performance indicators and then used this newly defined feature within logistic regression to model student risk [1]. This method proved effective with 80+ percent accuracy in predicting student dropout, leading to high popularity and wide-scale implementation and use [6].

More recently, researchers have begun to leverage machine learning and data mining methods, sometimes termed predictive analytics, to find complex patterns associated with future

Chad Coleman, Ryan Baker and Shonte Stephenson "A Better Cold-Start for Early Prediction of Student At-Risk Status in New School Districts" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 732 - 737

student outcomes [28, 17]. In K-12 education, this approach was used by Lakkaraju et al. [29] to predict student dropout in two districts, finding that the Random Forest algorithm outperformed several other algorithms. Some of the efforts to use machine learning in predicting student success have been scaled beyond single districts to entire states [27]. However, it still remains a challenge to deploy predictive analytics for use in schools at scale. District data often contain substantial information about its schools and students: demographic data about the student and teacher populations, academic performance information, financial information, disciplinary actions, and attendance records [36]. However, in many school districts, data quality is limited, with key information only available by integrating across multiple data warehouses, incompatible student ID numbers, errors in data entry, and local idiosyncratic interpretations of often ambiguous data fields. Even when data is today excellent, key data from past years is often unavailable due to the absence of a formal data system or having a data system which is difficult to query. Semantics may also change -- for example, the definition of "not graduated" is not stable across years and contexts [35] -- but these changes may not always be clearly understood when reviewing past data.

One solution is to use models that involve simple variables that are feasible for almost any district's data and assume that model will be valid in new contexts, even where that context may be quite different from the context where the model was initially developed [32]. The Chicago model [1] is a common choice for this type of application.

In this paper, we propose and evaluate an alternate solution to providing a model for a new district. Our approach attempts to generate predictions for a specific "Target" school district based on models from other school districts where full datasets are available, using a simple average of the district models, where all existing models are given equal weight. We compare the quality of our averaging approach to the earlier solution of using a simple generic model, specifically, the Chicago model.

2. METHODS

In the following section, we will discuss our method for making at-risk student predictions for school districts which have insufficient data to create a district-based prediction model. In brief, we develop and validate predictive analytics models for each school district with sufficient data. These models predict each student's probability of graduating (or risk of not graduating). We then conduct a simple ensembling approach, averaging each model's predictions, to produce a single prediction for each student. We test the quality of this approach by conducting it for held-out districts where data is available.

We validate our new approach by comparing its performance to the widely-used "Chicago model" [1, 6] on the same test data and comparing the performance of our detector to the classic Chicago model, which can be used for entirely new districts with no re-training. The Chicago model utilizes freshman-year GPA, the number of semester course failures, and freshman-year absences to determine the student's risk of failing to graduate [1]. Since the Chicago model relies on data collected within the first year of high school, we were only able to compare the performance of our approach to the Chicago model for high school students.

2.1 Data

Data for this research originate from the BrightBytes data analytics and visualization platform, Clarity®. The Clarity® platform ingests disparate datasets, transforms them to a standardized format by mapping district-specific variables to a common schema, prepares the data for analysis, and then visualizes the data in a meaningful, easy-to-understand way. The Clarity® platform is used by 1 in 5 schools across 47 states to empower educational leaders to use data for decision making. The value derived by districts from the Clarity® platform comes from using data to drive change within an organization. The anonymized dataset used in this paper ($n=3,575,724$) represents a large spectrum of K-12 students in terms of free/reduced lunch eligibility, school urbanicity, and school demographic makeup, and is drawn from a range of school districts, educational organizations and agencies.

We have nearly complete data (with only small numbers of variables unavailable) from an educational agency with decision-making power over a large geographical region (Pillar 1) and three large individual school districts (Pillar 2, Pillar 3, Pillar 4). These datasets are referred to as "Pillars" because they serve as bases for our ensemble-based approach. The four Pillars differ in terms of their predominant student demographic groups, with Caucasians representing the largest group of students in Pillar 1 ($n=1,681,988$), Hispanic/Latinx students representing the largest group of students in Pillar 2 ($n=392,148$), and split demographics in Pillar 3 ($n=158,991$) and Pillar 4 ($n=140,132$).

We test our models on 30 "Target" districts that were not used to develop the models, due to having fewer years of data, more missing variables, or smaller samples overall. These Target districts span a diverse range of predominant demographics, with one Target district being over ninety percent Caucasian at one extreme, and other districts being almost completely Hispanic/Latinx or African American. District performance is equally as diverse: some Target districts achieve graduation rates over 90% while others have graduation rates as low as 36%. Table 1 below highlights the number of records available in the Pillar districts and Target districts. The Target districts are generally smaller than the Pillar districts, with some having as few as 271 total historical student records, with the percent of data missingness within the Target districts also quite high in some cases ($M=41.65\%$, $SD=7.498\%$).

Table 1: Average Number of Outcome Records in Target Districts

Grade Band	Graduates	SD	Dropouts	SD
1st – 5th	4,307	9,784	634	1,267
6th – 8th	7,673	16,837	996	2,012
9th – 12th	24,552	39,524	1,920	3,833
All Grades	12,177	95,962	1,183	7,377

2.2 Predictor Variables

The potential set of predictor variables was selected in partnership with the American Institutes for Research (AIR) team [22], this paper's authors, and other researchers and developers at BrightBytes. This collaboration resulted in a theory-based [9] framework of success indicators, along with definitions of those success indicators that are used to map and align district data. Due to the data ingestion and transformation

process, the same data features can be used across all districts. Below is a distillation of the broad range of potential variables into a small set of meaningful buckets:

General Coursework: student academic performance such as total credits earned or student grade point averages [25, 38, 5].

Student Assessments: interim or summative assessments related to math, science, reading and social studies performance [26,16].

Student Attendance: recorded as absences or tardies [33].

Student Behavior: disciplinary incidents the student has on file [4, 10].

2.3 Model Fitting

The first step toward building an at-risk prediction model for districts without sufficient data is to build models for districts with sufficient data. For each of these models, we took the data from a single district. We filtered down to only the students who were flagged as ‘dropped’ and ‘graduated’. Even if students took extra time to graduate, they were still counted as graduating. Only these students were used for building the model; all other outcomes (such as transferring to another school district) were removed from the filtered dataset. The resultant datasets were generally highly imbalanced, with substantially more students graduating than dropping out. To account for this imbalance, the training data was manually rebalanced by adding duplicate copies of students who dropped out to the data set. Specifically, duplicates were created such that every grade level (10th, 11th, 12th, etc.) of students in the training datasets had an equal number of students who dropped out as students who remained. The original data distribution was used when testing the models.

Decision trees, support vector machines, XGBoost, logistic regression and random forest were all tested to build the initial model. The best performance across data was obtained with random forest classifiers with $n=15$ estimators and a max depth of 10. Since the algorithm was tree based, we utilized arbitrary value substitution to replace missing values with a high integer [39]. The goodness of each district’s model was evaluated, within-district, using a train-test split method (note that models are also evaluated within entirely new districts; see below). In each case, the training set consisted of a randomly selected 70 percent of the data with label-based stratification used across grades. The test set held out to validate the model consisted of the remaining 30 percent of the data.

Models were evaluated using the Area Under the Curve for the Receiver Operator Characteristic graph. AUC ROC was selected as our primary evaluation statistic due to its interpretability and validity for highly-imbalanced test sets [24]. AUC ROC calculates the tradeoff between true positive and false negative for every possible threshold used for labeling data points as positive and negative; as such, it is well-suited for evaluating how well an algorithm ranks students relative to their risk.

2.4 Pillar Selection

Selection of the four Pillar districts was based on two factors, data quality and model performance. To evaluate data quality, we calculated the proportion of missing values within the total feature set, expressed as a percentage. Districts were not included as Pillar districts if they had high amounts of missing data, over 40% of values missing, as these districts would be less useful for modeling other districts where these features were present. Districts were also not included as Pillar districts

if they lacked historical data spanning all grades 1 through 12; districts without historical data for some grades would be less useful for developing models that could be applied to all grades.

Models developed for specific districts as potential pillar model candidates were fit and evaluated using held-out test sets from that district’s own data. Districts for which we were able to produce a model with AUC higher than 0.7 on the district’s test sample, averaged across all student class years, were designated as Pillar districts/models and used in our predictions for districts for which models could not be generated for all grade levels, or for which models were insufficient in quality. Of the 30 Target districts within our study, 25 do not have enough historical records spanning all 12 grades and 5 had sufficient data but were unable to produce an AUC over 0.7. All 30 districts suffered from at least some degree of feature missingness.

2.5 Applying Models to New Districts

Having developed models for Pillar districts, where data are abundant, data quality is high, and where it is possible to develop a high-quality model, we next applied each Pillar model to each Target district. These Target districts had at least one of the following attributes; 1) Under 20,000 students, 2) Over 40% missing values, 3) Missing historical records for some grades in K-12, 4) AUC ROC when applied to new students within-district.

Our first step to applying the Pillar models was simply to run each of them on the Target district’s data ($n = 1,202,465$) and obtain predictions for each student. This provides us with a set of predictions for each student and for each model. We then took the average of the probability estimates, across districts, to generate the final student prediction. When we applied Pillar models to Target districts, we evaluated these models using all historical records present in the data as none of their records were used within model training. As with models tested on the district for which they were built, we use AUC ROC as our metric of model goodness.

3. RESULTS

3.1 Within-District Performance

We first applied each Pillar district model to new students from the same district, to evaluate within-district performance. As shown in Table 2, the four Pillar districts achieved AUC ROC values ranging between 0.899-0.936 when predicting graduation/dropout, for 9th through 12th grades (the typical high school years in U.S. classrooms). Performance was moderately lower for 6th-8th graders, where longitudinal predictions of up to 6 years are being made, with AUC ROC ranging from 0.849-0.884. Performance was again moderately lower for 1st-5th graders, where longitudinal predictions of up to 11 years are being made, with AUC ROC ranging from 0.758-0.810.

Table 2. AUC of Pillar Model Performance on Pillar Model Test Data (new students) by Grade Band

District	1 – 5	6 - 8	9 - 12	Average
Pillar 1	0.778	0.849	0.899	0.865
Pillar 2	0.758	0.884	0.908	0.858
Pillar 3	0.810	0.884	0.936	0.888

Pillar 4	0.794	0.850	0.903	0.886
----------	-------	-------	-------	-------

Our attempts to build a model for each Target district proved less successful, with 9th – 12th grade model AUC averaging 0.729, 0.1825 lower than for the Pillar Districts, 6th-8th model AUC averaging 0.669, 0.198 AUC lower than the Pillar Districts, and 1st -5th grade obtaining an average AUC of 0.635, 0.15 points lower than the Pillar Districts' within-district performance for these grade levels.

It should be noted that there were three exceptions to this poor AUC trend. Three Target districts produced relatively more successful models, averaging AUC = 0.81 for 9th-12th grade students, AUC = 0.709 for 6th-8th grade students, and AUC = 0.666 for 1st-5th grade students. Despite the initial successful results of these three Target districts, they still lacked the data to produce models for all years, with all three missing records for 1st and 2nd grades. When additional historical records become available in the future, these models will almost certainly make it into the pool of Pillar models in future model iterations.

3.2 Feature Importance

We can understand which features are particularly important to each Pillar model by calculating feature importance. Feature importance was calculated using the mean decrease impurity method, sometimes referred to as the *gini importance* [12]. This metric allows us to calculate how much each feature contributes to the model's eventual predictions of a student's outcomes (in this case, risk of dropout). A range of different types of features were found to be important in the four models. The Pillar 1 model relied heavily on features related to student age, attendance and academic achievement. The Pillar 2 model was similar to the Pillar 1 model in that it relied strongly on academic and attendance related features. However, student summative reading scores were also important to the Pillar 2 model. The Pillar 3 model was less similar to the first two Pillars. For Pillar 3, student assessment data and student behavioral data (i.e., disruption, defiance, etc.) were the primary contributors to the model's predicted outcomes, a difference in feature importance that is likely due to a multitude of reasons. Pillar 4 was most similar to Pillar 1 as it also relied heavily on student academic indicators. One reason could be that there were differences in the data availability of features for each district. For example, no assessment data was available for Pillar 1, whereas Pillar 3 had assessment data available for almost all of their historical student records. Another cause could be the difference in the populations of students in each Pillar district. For example, attendance may play a larger role in graduation in urban districts (e.g. Pillar 2), whereas behavioral incidents could play a larger role in the path to dropping out for students in more rural districts (e.g. Pillar 3).

3.3 Performance on New Districts

We applied the Pillar models to each student's data from the 30 Target districts, and averaged the probability across models for each student. These districts had considerable variation in size, graduation rate, and degree of missingness of data (and which features were missing), with values for these variables that were substantially higher or lower than the values for the Pillar districts. In other words, applying models from the Pillar districts to these thirty Target districts represents substantial extrapolation.

Table 3 shows average performance of each individual Pillar model detectors on the Target district data, as well as using averaged probabilities. Despite the high degree of extrapolation required, performance was generally good, with an average AUC (across districts) of 0.783 ($SD = 0.100$), with three districts achieving AUC above 0.9. AUC results within grade band produced similar outcomes, with 9th-12th obtaining an average AUC of 0.813 ($SD = 0.078$), 6th-8th model AUC averaging 0.736 ($SD = 0.13$), and 1st -5th grade model AUC averaging 0.646 ($SD = 0.141$). However, two districts (Target 12 and Target 28) had poor overall AUC values of 0.539 and 0.469. It is worth noting that these two districts had the highest rate of missing data for features that ranked most important in the Pillar models, with over 80% of students in these Target districts missing data related to coursework, over 90% of the records not containing any assessment scores, and the data for 40% of the students not containing attendance information. Overall, the districts with the highest amounts of missing data in core features were also the districts with the lowest AUC ROC values. None of the individual Pillar models did as well as their average when applied to the Target districts; individual Pillar models achieved an AUC between 0.718 - 0.756 on the Target districts, significantly underperforming compared to averaging the produced model probabilities.

Table 3: Average Performance of Pillar Model and Mean Detectors on Target District Data.

Detector	1 – 5	6 - 8	9 - 12	All Grades
Mean Model	0.646	0.74	0.813	0.783
Pillar 1 Model	0.631	0.673	0.749	0.719
Pillar 2 Model	0.568	0.678	0.764	0.718
Pillar 3 Model	0.631	0.720	0.789	0.750
Pillar 4 Model	0.591	0.685	0.788	0.756

3.4 The Chicago Model On-Task Indicator

Comparing our Mean Model detector to the Chicago model detector was limited by data availability: the Chicago model relies on freshman year high school student data, specifically the number of course credits and courses failed during freshman year (9th grade). Due to the detector's reliance on these two data points, our validation sample was limited to only those districts that contained valid information for these two features. However, many of our Target districts lacked data for the features in the Chicago model, for some students. If at least one feature was available for the Chicago model, the model was used; a student was assigned a .5 probability of graduating if the Chicago model was missing all features and therefore incapable of producing a prediction. The Pillar models performed also poorly for these students with very high data missingness.

Table 4: Average AUC Performance of Mean Model vs. Chicago On-Task Indicator: 9th-12th grades

Detector	Avg AUC	Standard Deviation
Mean Model	0.821	0.084
Chicago Model	0.624	0.121

The Mean Model outperformed the Chicago model in every Target district, except for one district. In that district, the Chicago model ($AUC=0.77$) performed .068 better than the Mean Model ($AUC=0.702$). Overall, the mean Pillar model

detector achieved an average AUC of 0.197 higher than the Chicago model when measuring performance of predictions on high school student Target district data.

One might argue that this comparison biases against the Chicago model, by including cases where the specific data needed was unavailable. Although our approach is designed to work in cases of high missingness, we can also compare our Mean Model to the Chicago Model only for cases which have the data the Chicago model needs. This resulted in a significant reduction of the data used to calculate AUC metrics, with only 351,902 out of the original 1,202,465 student records able to populate the On-Track Indicator, 29.3% of our initial sample.

Table 5: AUC Performance of Mean Model vs. Chicago On-Track Indicator on Complete Records: 9th-12th grades

Detector	Avg AUC	Standard Deviation
Mean Model	0.874	0.061
Chicago Model	0.734	0.082

Both the Mean Model and the Chicago Model saw an increase in their average model performance across the new sample, with the Mean Model increasing by 0.053 from 0.821 to 0.874 and the Chicago model increasing by 0.11, from 0.624 to a more respectable AUC of 0.734. However, the Mean Model still achieves an AUC 0.14 higher despite these conditions designed to be more favorable to the Chicago Model.

4. CONCLUSIONS

In this paper we propose an approach to predict student risk of not graduating from high school for districts where the quality, quantity, or availability of data is insufficient to produce a satisfactory model of student risk, using an ensemble of models from other districts where data is available. This method achieves good predictive power for students in districts that were not used to develop the model, without any fitting or modification to the models or their application. Furthermore, it achieves substantially better results than a popular alternate approach to predicting at-risk status in new districts, the Chicago model.

It is worth noting that our approach and study have several limitations that should be investigated in future work. Though our sample of Target districts was large, we have not yet applied this method across the full diversity of students in the U.S. In particular, districts with substantial Native American populations or those located in extremely rural regions, such as northern or western Alaska, are not represented in our study. Similarly, we have not studied whether our models are equally good for all subgroups within the school districts—a limitation that is common in the field.

There are several ways in which we could probably improve model performance. Research has shown that contextual factors can contribute to identifying students at risk of dropping out and that factors associated with dropout can differ between populations [7, 15]. Altering the detector to weight the Pillar model probabilities by leveraging characteristic information such as student and school demographics, urbanicity (urban, rural, suburban), and the proportion of military-connected or otherwise highly-mobile students could help account for similarities between students and districts better. Additionally, future iterations of our method could take an empirical approach

to selecting the Pillar model weights using measure of similarity based on model performance [31], rather than limiting the approach to the current simple averaging method where features are weighted equally.

Ultimately, the performance of the Mean Model presents new opportunities in identifying students at risk of dropping out for districts with minimal or no data. Given the potential benefit of interventions for at-risk students, this new approach has the capacity to improve the future of student outcomes within a large number of schools where it is not yet possible to develop predictive models. Students educated by districts where data are insufficient can now be presented with greater opportunities through the use of proactive interventions driven by predictive modeling rather than being limited to receiving reactive interventions that are often applied too late, if ever.

5. REFERENCES

- [1] Allensworth, E. M., & Easton, J. Q. (2007). What Matters for Staying On-Track and Graduating in Chicago Public High Schools: A Close Look at Course Grades, Failures, and Attendance in the Freshman Year. Research Report. Consortium on Chicago School Research.
- [2] Amos, Jason (2008). *Dropouts, Diplomas, and Dollars: US High Schools and the Nation's Economy*. Washington, DC: Alliance for Excellent Education.
- [3] Baker, R. S., & Yacef, K. (2009). The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining*, 1(1), 3-17.
- [4] Balfanz, R., Byrnes, V., & Fox, J. H. (2015). Sent home and put off track: The antecedents, disproportionalities, and consequences of being suspended in the 9th grade. In D. J. Losen (Ed.), *Closing the School Discipline Gap* (pp. 17-30). New York, NY: Teachers College Press.
- [5] Balfanz, R., DePaoli, J. L., Ingram, E. S., Bridgeland, J. M., & Fox, J. H. (2016). *Closing the College Gap: A Roadmap to Postsecondary Readiness and Attainment*. *Civic Enterprises*.
- [6] Balfanz, R., Herzog, L., & Mac Iver, D. J. (2007). Preventing student disengagement and keeping students on the graduation path in urban middle-grades schools: Early identification and effective interventions. *Educational Psychologist*, 42(4), 223-235.
- [7] Balfanz, R., & Legters, N. (2004). *Locating the Dropout Crisis. Which High Schools Produce the Nation's Dropouts? Where Are They Located? Who Attends Them? Report 70*. Center for Research on the Education of Students Placed at Risk CRESPAR.
- [8] Belfield, C. R., & Levin, H. M. (2007). *The price we pay: Economic and social consequences of inadequate education*. Washington, D.C.: Brookings Institution Press.
- [9] Bernhardt, V. L., & Bernhardt, V. (2013). *Data analysis for continuous school improvement*. Routledge.
- [10] Bowers, A. J., Spratt, R., & Taff, S. A. (2012). Do we know who will drop out? A review of the predictors of dropping out of high school: Precision, sensitivity, and specificity. *The High School Journal*, 77-100.
- [11] Bowers, A.J., & Spratt, R. (2012). Examining the multiple trajectories associated with dropping out of high school: A

- growth mixture model analysis. *Journal of Educational Research*, 105(3), 176-195.
- [12] Breiman, L., & Cutler, A. (2007). Random forests-classification description. Department of Statistics, Berkeley, 2.
- [13] Cha, S. H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *International J. Mathematical Models and Methods in Applied Science*, vol. 1, no. 4, (pp. 300–307).
- [14] Christenson, Sandra L., and Martha L. Thurlow (2004). School Dropouts: Prevention Considerations, Interventions, and Challenges. *Current Directions in Psychological Science* 13(1): 36–39.
- [15] Christle, C. A., Jolivette, K., & Nelson, C. M. (2007). School characteristics related to high school dropout rates. *Remedial and Special education*, 28(6), 325-339.
- [16] Cumpton, G., Schexnayder, D., & King, C. T. (2012). Factors associated with education and work after high school for the classes of 2008 and 2009: A research report of the Central Texas Student Futures Project.
- [17] Dekker, G. W., Pechenizkiy, M., & Vleeshouwers, J. M. (2009). Predicting Students Drop Out: A Case Study. *International Working Group on Educational Data Mining*. Paper presented at the International Conference on Educational Data Mining (EDM) (2nd, Cordoba, Spain, Jul 1-3, 2009)
- [18] DePaoli, Jennifer L., Joanna Hornig Fox, Erin S. Ingram, et al. (2015). Building a Grad Nation: Progress and Challenge in Ending the High School Dropout Epidemic. Annual Update 2015. Civic Enterprises.
- [19] Driscoll, Anne K. (1999). Risk of High School Dropout among Immigrant and Native Hispanic Youth. *International Migration Review* 33(4): 857–875.
- [20] Enslinger, M. E., & Slusarcick, A. L. (1992). Paths to high school graduation or dropout: A longitudinal study of a first-grade cohort. *Sociology of education*, 95-113.
- [21] Finn, Jeremy D. (1989). Withdrawing from School. *Review of Educational Research* 59(2): 117–142.
- [22] Heppen, J. B., & Theriault, S. B. (2008). Developing Early Warning Systems to Identify Potential High School Dropouts. Issue Brief. National High School Center.
- [23] Hoffman, Nancy, Joel Vargas, Andrea Venezia, and Marc S. Miller (2007). Minding the Gap: Why Integrating High School with College Makes Sense and How to Do It. ERIC.
- [24] Jeni, L. A., Cohn, J. F., & De La Torre, F. (2013). Facing Imbalanced Data--Recommendations for the Use of Performance Metrics. *Proceedings of the International Conference on Affective Computing and Intelligent Interaction*, 245-251.
- [25] Kemple, J. J., Segeritz, M. D., & Stephenson, N. (2013). Building on-track indicators for high school graduation and college readiness: Evidence from New York City. *Journal of Education for Students Placed at Risk (JESPAR)*, 18(1), 7-28.
- [26] Koon, S., & Petscher, Y. (2016). Stated briefly: Can scores on an interim high school reading assessment accurately predict low performance on college readiness exams?
- [27] Knowles, J. E. (2015). Of needles and haystacks: Building an accurate statewide dropout early warning system in Wisconsin. *Journal of Educational Data Mining*, 7(3), 18-67.
- [28] Kotsiantis, S. B., Pierrakeas, C. J., & Pintelas, P. E. (2003). Preventing student dropout in distance learning using machine learning techniques. *Proceedings of the International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 267-274.
- [29] Lakkaraju, H., Aguiar, E., Shan, C., Miller, D., Bhanpuri, N., Ghani, R., & Addison, K. L. (2015). A machine learning framework to identify students at risk of adverse academic outcomes. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1909-1918). ACM.
- [30] Legault, Lisa, Isabelle Green-Demers, and Luc Pelletier (2006). Why Do High School Students Lack Motivation in the Classroom? Toward an Understanding of Academic Amotivation and the Role of Social Support. *Journal of Educational Psychology* 98(3): 567.
- [31] McCune, B., Grace, J. B., & Urban, D. L. (2002). Analysis of ecological communities (Vol. 28). Glenden Beach, OR: MjM software design.
- [32] Neild, R. C., Stoner-Eby, S., & Furstenberg, F. (2008). Connecting entrance and departure: The transition to ninth grade and high school dropout. *Education and Urban Society*, 40(5), 543-569.
- [33] Peugh, J. L., & Enders, C. K. (2004). Missing data in educational research: A review of reporting practices and suggestions for improvement. *Review of educational research*, 74(4), 525-556.
- [34] Rafa, A. (2017). Chronic absenteeism: A key indicator of student success. Policy Analysis. Education Commission of the States. Retrieved from <https://files.eric.ed.gov/fulltext/ED574526.pdf>
- [35] Rumberger, R. W. (1987). High school dropouts: A review of issues and evidence. *Review of educational research*, 57(2), 101-121.
- [36] Schildkamp, K., Lai, M. K., & Earl, L. (Eds.). (2012). Data-based decision making in education: Challenges and opportunities (Vol. 17). Springer Science & Business Media.
- [37] Stark, Patrick, and Amber M. Noel (2015). Trends in High School Dropout and Completion Rates in the United States: 1972-2012. Compendium Report. NCES 2015-015. National Center for Education Statistics.
- [38] Stuit, D., O'Cummings, M., Norbury, H., Heppen, J., Dhillon, S., Lindsay, J., & Zhu, B. (2016). Identifying early warning indicators in three Ohio school districts. National Center for Education Evaluation and Regional Assistance.
- [39] Venables, W. N., & Ripley, B. D. (2002). Tree-based methods. In *Modern Applied Statistics with S* (pp. 251-269). Springer, New York, NY.

Measuring Item Teaching Value in an Online Learning Environment

Jon Harmon
Macmillan Learning
211 E. 7th St.
Austin, TX 78701
+1 512-323-6565

jon.harmon@macmillan.com

Rasil Warnakulasooriya
Macmillan Learning
75 Arlington St.
Boston, MA 02116
+1 617-399-4531
rasilw@gmail.com

ABSTRACT

The Additive Factor Model (AFM) is a cognitive diagnostic model that can be used to predict student performance on items in a context that allows for student learning. Within AFM, *skills* have a learning rate, and student acquisition of a skill depends only on the number of opportunities a student has had to exercise that skill and the learning rate of that skill. Here we demonstrate an approach to measure the *teaching value* of individual *items* with respect to one another. The teaching values estimated through this approach may be useful for structuring intelligent tutoring systems and for content improvement.

Keywords

cognitive diagnostic models, item response theory, knowledge tracing, data mining, content pedagogy, content improvement, priming.

1. INTRODUCTION

Item Response Theory (IRT) models describe the performance of students with respect to a set of scored items (questions). As described by Sijsma and Junker [5], most IRT models have three assumptions:

1. Local independence: Student performance on a given item does not depend on student performance on previous items.
2. Monotonicity: The probability of student success on an item increases when student ability improves.
3. Unidimensionality: Each student has the same ability for every item, and each item has the same difficulty for every student.

The simplest IRT equation, the 1-parameter logistic model or 1PL, can be expressed as

$$P(Y_{ij} = 1 | \alpha_i, \beta_j) = \frac{e^{\alpha_i - \beta_j}}{1 + e^{\alpha_i - \beta_j}} = f(\alpha_i - \beta_j)$$

where $f(x) = 1/(1 + e^{-x})$ (resulting in a probability in $(0, 1)$), Y_{ij} is the response of student i on item j (with 1 for correct, 0 for

incorrect), α_i is the ability of student i , and β_j is the difficulty of item j .

Multidimensional IRT (MIRT) models relax the third assumption, decomposing student abilities and/or item difficulties into an array of abilities or difficulties. The Additive Factor Model (AFM) proposed by Cen et al [2] can be viewed as a MIRT model that also relaxes the local independence assumption, taking into account multiple exposures to a skill through a learning rate for that skill. The probability of student success is expressed as:

$$P(Y_{ij} = 1 | \alpha_i, \beta, \gamma) = f\left(\alpha_i + \sum_{k=1}^K \beta_k q_{jk} + \sum_{k=1}^K \gamma_k q_{jk} t_{ik}\right)$$

where $f(x)$ is again the logistic function $1/(1 + e^{-x})$, β_k is the difficulty (or easiness, with the sign reversed) of *skill* (rather than item) k , q_{jk} is a binary indicator that item j uses skill k , γ_k is the learning rate of skill k , t_{ik} is the number of exposures student i has had to skill k , and K is the total number of skills assessed. If each item addresses only one skill, the first sum reduces to β_j . If the learning rate is 0, the second sum reduces to 0, resulting in the 1PL equation.

In AFM, the likelihood of a student acquiring a skill is impacted only by the constant learning rate of that skill and the number of exposures that student has had to that skill. Here we seek to estimate a new quantity, the *teaching value* of an item with feedback. Exposure of a student to an item with a positive teaching value increases the probability that that student will answer a subsequent question correctly.

2. METHODOLOGY

Macmillan Learning's homework system examined in this study serves items (questions with automatic grading and feedback) to students, with the items grouped into assignments. The items investigated in this study were used across 570 institutions of higher learning in 4,704 courses. The order of the items in the assignments could be partially or fully randomized, and could be edited manually by individual instructors. As a result, a given item could be the first item in an assignment for some students, the second for others, etc., even within the same course.

We began with a dataset containing 240,990 unique students interacting with 7,257 unique general chemistry items, resulting in 29,005,495 student-item interactions. To simplify this proof of concept, we assume that all questions measured a single skill ("general chemistry knowledge"). To measure the impact of questions on one another, we define an *experience* as a set of one, two, or three items presented to a student in a particular order, starting with the first item in the assignment. For each experience,

Jon Harmon and Rasil Warnakulasooriya "Measuring Item Teaching Value in an Online Learning Environment" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 738 - 741

we scored the experience as *correct* (1) iff the student was scored as correct on their first attempt on the *last* item in that experience, and *incorrect* (0) if they were scored as incorrect on their first attempt on the *last* item in that experience, regardless of how they performed on any previous items. Using the dplyr package [7] for the R programming language [3], we filtered the full dataset to 32,199 unique students, 1,264 unique one-item experiences, 1,956 unique two-item experiences, and 1,567 unique three-item experiences, resulting in 3,240,791 student-experience interactions, using the following conditions:

1. The experience contained one, two, or three items.
2. Every item within each experience was attempted by at least 100 students as the *first* item in their assignment.
3. Every experience was attempted by at least 100 students.
4. Each student attempted at least 50 experiences.

By treating each experience as a single item, we were able to model the student abilities and difficulties of questions using a relatively simple IRT model. In this study we used the two-parameter logistic model (2PL) described by Birnbaum [1]. In the 2PL, items are allowed to have varying difficulties (β) and discrimination values (the ability of an item to differentiate between a low-skill student and a high-skill student, herein denoted a). The probability (P) of a student with ability θ answering a question correctly is given by:

$$\ln\left(\frac{P}{1-P}\right) = a(\theta - \beta)$$

For the 2PL, it is assumed that the chance of a student *guessing* the correct answer is 0. The item and student parameters can be estimated using marginal maximum likelihood estimation (MML). Here we used the TAM package [4] to perform the estimations.

We fit the filtered student-experience interactions to a single two-parameter logistic model. To illustrate the approach in detail, our focus in this paper will be on the two-item experiences.

We define the *raw difficulty* (β_A) of a given item A as the modeled difficulty of item A when it is the *first* item attempted by students (i.e., when it is presented in a one-item experience). We define the *apparent difficulty* ($\beta_{A|B}$) of a given item A with respect to another item B as the modeled difficulty of item A when it appears *second* in a two-item experience, after item B. We calculated the *difficulty change* of a given item A after item B as the difference between the apparent difficulty and the raw difficulty for that item in that experience.

$$\Delta\beta_{A|B} = \beta_{A|B} - \beta_A$$

A *negative* difficulty change indicates that item A appears to be *easier* when it follows item B, and a *positive* difficulty change indicates that item A appears to be *harder* when it follows item B. For this study, we did not constrain the time between a student answering item A and that student answering item B.

3. RESULTS

Two "goodness of fit" statistics were used to evaluate the fit of each experience to the model: outfit and infit. Both measures are expected to have a value close to 1.0 for each experience if the model fits the data without overfitting or underfitting. 92% of experiences had outfit of 1.0 ± 0.05 (standard deviation), and 99% of experiences had infit of 1.0 ± 0.05 (standard deviation). We analyzed two-item experiences which met these criteria: 1) the

experience had infit and outfit between 0.95 and 1.05, and 2) the second item of the experience had infit and outfit between 0.95 and 1.05 when it was in a one-item experience. Seventy six percent (1,490 of 1,956) of two-item experiences met these criteria.

The median difficulty change for these experiences was found to be -0.085 , with the second quartile beginning at -0.42 , and the third quartile ending at 0.27 . We focused our analysis on experiences with difficulty changes more than 1.5 interquartile ranges above the third quartile or below the second quartile, and defined these difficulty changes as significant. Other difficulty changes may also be statistically significant, leading to net learning effects. However, the educational significance of such effects remains to be explored in future studies. We observed that 22 item pairs (1.5%) had a difficulty change more than 1.5 times the interquartile range below the first quartile, indicating a significant *decrease* in apparent difficulty (the items appeared to become significantly *easier*), as we expected would occur. Somewhat surprisingly, 17 item pairs (1.1%) had a difficulty change more than 1.5 times the interquartile range above the third quartile, indicating a significant *increase* in apparent difficulty (the items appeared to become significantly *harder*) (see Figure 1¹). The remaining 1451 item pairs (97.4%) did not have a significant change in apparent difficulty as defined here. We examined specific cases of each type of change in apparent difficulty to attempt to identify the sources of the difficulty changes.

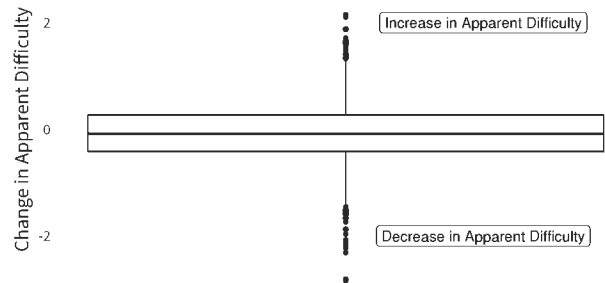


Figure 1. Most difficulty changes (the second and third quartile, shown in the box here) were near zero. However, some items showed a decrease in apparent difficulty (points at the bottom of the plot, lying more than 1.5 times the interquartile range below the second quartile), and some items showed an increase in apparent difficulty (points at the top of the plot, lying more than 1.5 times the interquartile range above the third quartile).

3.1 Decrease in Apparent Difficulty

Here we examine a specific case of difficulty changes in which the second item appears to be *easier* for students who have been primed by a specific preceding item. The item we designate "item A" in this study asks the student to

Give the conjugate acid for each compound below.

with three randomly selected bases such as HSO_4^- , CO_3^{2-} , and NH_3 . The student must enter the conjugate acid for each base (in this example, H_2SO_4 , HCO_3^- , and NH_4^+). The item we designate "item B" is effectively the opposite question, in which the student is given three acids, and asked to enter the conjugate base. Even when the

¹ All figures were generated using the ggplot2 package for R [6].

randomly selected acids and bases do not line up from item B to item A for a given student, students perform significantly better when primed by item B before answering item A, or *vice versa*.

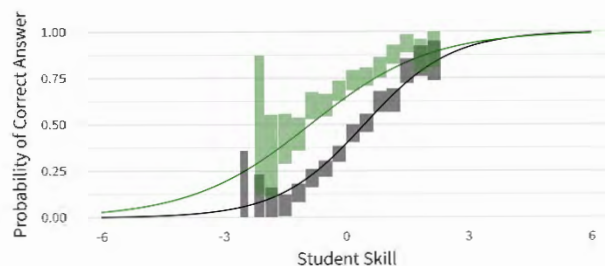


Figure 2. When a specific item A followed a specific item B (green), students performed better than when item A was the first question (gray); the curve shifted up and to the left, indicating that students with lower skill became more likely to answer the question correctly. The curves indicate the modeled difficulty and discrimination for each experience, while the boxes indicate the 95% confidence interval of actual student performance (% of students in a given modeled skill group who scored correct on the experience on their first try). See Figure 3 for the same items in the opposite order.

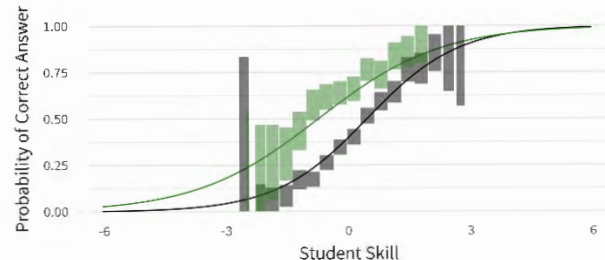


Figure 3. When item B followed item A (green), students performed better than when item B was the first question (gray). See Figure 2 for the same items in the opposite order.

In contrast, the item we designate "item C" is within the same general topic, asking the student to

Label each reactant and product in this reaction as a Brønsted acid or base.

and showing the reaction between HCN and NH_2^- to produce CN^- and NH_3 . Item C does *not* have a significant impact on student performance on item A (data was not available for student performance on item B after item C).

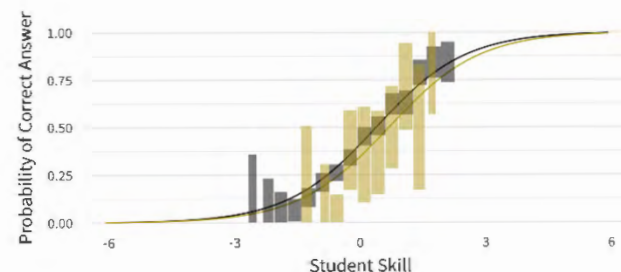


Figure 4. When item A from Figures 2 and 3 followed a third specific item C (yellow), there was no significant change from when item A was the first question (gray).

The raw difficulty for item A was found to be 0.45 ± 0.05 (95% confidence interval reported on all fit difficulties). When item A followed item B, the apparent difficulty was -0.9 ± 0.1 , a difficulty change of -1.4 ± 0.1 (see Figure 2). Similarly, the raw difficulty for

item B was 0.5 ± 0.1 and the apparent difficulty after item A was -0.7 ± 0.1 , resulting in a difficulty change of -1.2 ± 0.1 (see Figure 3). In contrast, when item A followed item C, the apparent difficulty was not significantly different from the raw difficulty (0.7 ± 0.4 vs 0.45 ± 0.05) (see Figure 4).

3.2 Increase in Apparent Difficulty

Here we examine a specific case of difficulty changes in which the second item appears to be *harder* for students who have been primed by a specific preceding item. The item we designate "item D" asks,

Parts per million (ppm) is a common way to express small concentrations of a solute in water. A sample of tap water that is 25 ppm Cl^- contains 25 grams of Cl^- for every 1,000,000 grams of water. Which units are numerically equal to ppm for dilute aqueous solutions?

Students are given 5 choices in a random order: "g/L", "cg/L", "mg/L", "μg/L", and "ng/L". The correct answer is "mg/L".

A possible explanation for this effect can be found by examining the terminology used in each item, and the particular choices that students selected. Item E asks students to

Match each term with its definition or description.

with eight terms, including "parts per million." In the item we designate "item E," parts per million is defined as "micrograms of analyte per gram (or mL) of sample." In other words, Item D defines ppm in terms of L ("mg/L"), and item E defines ppm in terms of mL ("μg/mL"). Students who answer item E before item D are more likely to answer item D incorrectly, and they are more likely to do so by choosing the incorrect answer "μg/L" (see Figure 5). These results imply that students were led astray by the similarity of numerators of the units.

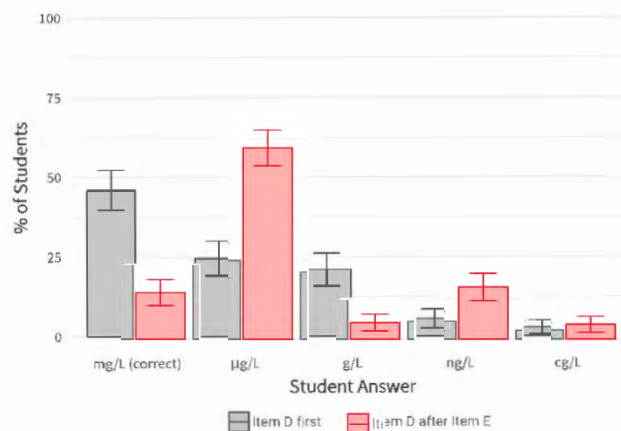


Figure 5. When item D followed item E (red), students chose "μg/L" more often than when item D was the first question (gray).

The raw difficulty of item D was 0.2 ± 0.3 . When item D followed item E, the apparent difficulty was 1.9 ± 0.4 , a difficulty change of 1.7 ± 0.4 (see Figure 6).

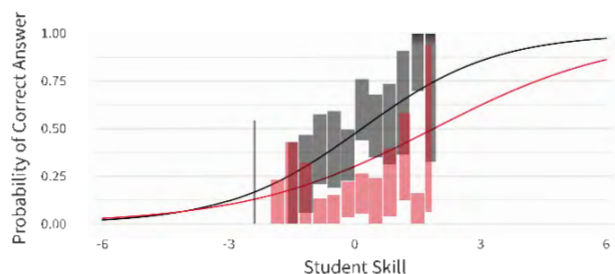


Figure 6. When item D followed item E (red), students answered incorrectly more often than when item D was the first question (gray); the curve shifted down and to the right, indicating that students required higher skill to have an equal probability of answering correctly.

4. CONCLUSION

We have shown that, adopting the methodology described above, we can measure the impact of individual questions with feedback on the difficulty of subsequent questions. This approach has implications for developing and revising pedagogically sound content. This approach could also influence sequencing of content to reinforce learning. Further research is required to attempt to generalize this difficulty change into a per-item “teaching value” parameter, but there appears to be evidence for this approach.

Items can have both positive and negative priming effects on the apparent difficulties of other items. Further research is required to determine the extent to which this effect increases or decreases with greater time in-between items, and whether intervening items have an impact on this effect.

5. ACKNOWLEDGEMENTS

JH would like to thank Jonathan Bratt for serving as a sounding board as this process was developed and for continued feedback throughout the process. JH and RW thank Adam Black for the opportunity to conduct this study.

6. REFERENCES

- [1] Birnbaum, A. 1968. Some latent trait models and their use in inferring an examinee’s ability. In F. M. Lord & M. R. Novick, *Statistical Theories of Mental Test Scores*. 397-472. Reading, MA: Addison-Wesley Publishing.
- [2] Cen, H., Koedinger, K.R., and Junker, B., 2006. Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. . In *Intelligent Tutoring Systems: 8th International Conference, ITS 2006, Zhongli, Taiwan, June 26-30, 2006. Proceedings*, M. Ikeda, K. D. Ashley, and T.-W. Chan, Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 164–175.
- [3] R Core Team. 2017. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.
- [4] Robitzsch, A., Kiefer, T., and Wu, M. 2018. *TAM: Test Analysis Modules*. <https://CRAN.R-project.org/package=TAM>.
- [5] Sijtsma, J., and Junker, B.W., 2006. Item response theory: Past performance, present developments, and future expectations. *Behaviormetrika*, 33(1), 75-102.
- [6] Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. <http://ggplot2.org>.
- [7] Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2018. *Dplyr: A Grammar of Data Manipulation*. <https://CRAN.R-project.org/package=dplyr>.

Detecting Outlier Behaviors in Student Progress Trajectories Using a Repeated Fuzzy Clustering Approach

Colm P. Howlin
Realizeit
Dublin, Ireland
colm.howlin@realizeitlearning.com

Charles D. Dziuban
University of Central Florida
Orlando, FL
charles.dziuban@ucf.edu

ABSTRACT

Clustering of educational data allows similar students to be grouped, in either crisp or fuzzy sets, based on their similarities. Standard approaches are well suited to identifying common student behaviors; however, by design, they put much less emphasis on less common behaviors or outliers. The approach presented in this paper employs fuzzing clustering in the identification of these outlier behaviors. The algorithm is an iterative one, where clustering is applied, outliers identified, the data restricted to the outliers, and the process repeated. This approach produces a clustering that is crisp between each iteration and fuzzy within. It arose as a consequence of trying to cluster student progress trajectories in an adaptive learning platform. Included are results from applying the repeated fuzzy clustering algorithm to data from multiple courses and semesters at the University of Central Florida, (N=5,044).

1. INTRODUCTION

Personalization holds the promise of making learning more engaging and effective for students. Each student can receive personalized feedback and guidance based on their interaction with the learning material and their current needs and goals. Key to being able to provide this is an understanding of the full range of learning behaviors that students can exhibit, and the driving forces behind them. Truly personalized learning needs to understand not just the most common behaviors, but also those that are more atypical or outliers.

A variety of techniques have been employed to uncover student behaviors in different learning contexts [22]. Clustering is a common approach with a considerable range in both the applications and the algorithm employed [25]. Applications have included adapting question delivery, promoting group-based collaboration, and the characterization of atypical student behavior.

This work presents a clustering approach to automatically detect and quantify the range of behaviors, including the

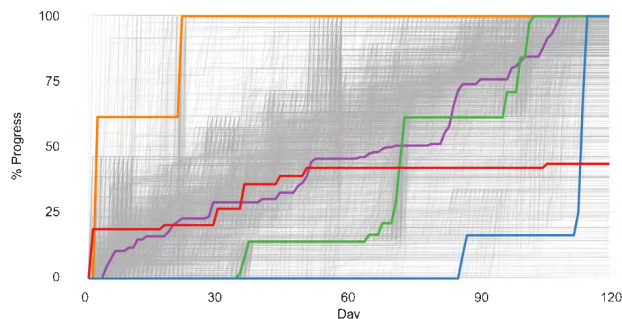


Figure 1: Student progress trajectories. The gray lines show the trajectories for all 5,044 students at UCF. The colored lines highlight several individual trajectories.

outliers, that are evident in student progress data, in order to provide feedback to instructors on their student's behaviors. This goal throws up two restrictions on our approach. First, the clustering of behaviors must be fully automated. Not all instructors will have the required knowledge to make decisions such as picking the parameters of the clustering algorithm. Therefore these decisions need to be handled by the algorithm. Second, the output from the clustering must be readily interpretable by an instructor, including both understanding what makes a cluster a cluster, but also easily understand the differences between clusters. These two restrictions provide a means of measuring the ultimate effectiveness of the algorithm and the quality of the clusters that it produces.

In [11], the authors examined student progress data against time for an online course delivered at the University of Central Florida (UCF) through the Realizeit adaptive learning platform. The course was self-paced with students free to set their rate of progress. While most set a steady, consistent pace over the 15-week term, some students set a very different pace. These outliers roughly fall into two categories: students who race ahead of the rest, and those who fall behind, leaving all their learning to the last minute.

Figure 1 provides an understanding of the challenges when clustering these progress trajectories. The x-axis represents time in days, and the y-axis is progress measured as the percentage of concepts mastered. The progress trajectories for 5,044 students across 51 online course instances in 9 terms

Colm Howlin and Charles Dziuban "Detecting Outlier Behaviors in Student Progress Trajectories Using a Repeated Fuzzy Clustering Approach" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 742 - 747

at UCF are shown in gray. Each line represents a single student. Patterns are difficult to distinguish, but the consistent trajectory of most students through their course is evident along the diagonal. Five progress trajectories (colored lines) have been singled out to highlight the range of possible behaviors. The challenge in clustering this data comes from the fact that clustering algorithms, by design, attempt to group the data into as few clusters as possible and therefore put much less emphasis on outliers. They seek the most common patterns. Our goal is to find both common and outlier behaviors.

Our approach draws inspiration from He et al. [16, 17] who used clustering to search for hidden communities in social networks. In their work, they first used clustering to discover the most apparent communities. They then decreased the weights on the edges in the social network that represented these communities. Repeating the clustering uncovers previously hidden communities. Our approach, repeated fuzzy clustering (RFC) uses a similar technique where clustering is applied, outliers identified, the data restricted to the outliers, and the process repeated. The purpose of this paper is to describe and demonstrate the RFC algorithm.

Algorithmic clustering methods are essentially “blind” in that there is no linguistic functioning in their process. The categories identified are impervious to shared characteristics that ground themselves in cultural beliefs. However, the linguistic and algorithmic categorization processes do have common intersections. Linguistically, Rosch, [23, 24], described this as prototype theory where through any number of cultural and societal processes what is the best representational icon of a category is formed by our preconceived notions. Adaptive learning provides diverse paths to success, many of which may not align with our preconceived notions of what constitutes successful or unsuccessful behavior. Clearly, clustering algorithms have assumptions built into them a priori but once built are not influenced by preconceptions. The questions we ultimately wish to address involves whether or not the clustering of student trajectories can provide a foundation for category characteristics through the multiple lenses of methods, education, linguistics, and prototype theory and should they make educational sense how can we use them to improve learning? [20].

2. APPROACH

Here we provide an outline of the RFC algorithm. In the following subsections, we provide the specifics on our implementation of each function, although it is possible to alter these to suit other needs or implementations. The algorithm proceeds by first grouping students using fuzzy clustering for a range of values of k (the number of clusters) - lines 5 \rightarrow 7. Validity indices are calculated for each solution, and the most appropriate number of clusters is chosen - line 8. The algorithm then proceeds by identifying outliers and removing them from the data. The algorithm then reapplies the clustering creating a more compact solution. This part of the process repeats until the algorithm identifies no new outliers - line 10. The data is then limited to the previously identified outliers on this loop - lines 11 \rightarrow 12. The whole process then repeats with the data filtered to the outliers.

There are three parameters to the algorithm: k_{max} is the

Algorithm Repeated Fuzzy Clustering

```

1:  $D$  is the student data
2:  $Outliers = All\ students$ 
3:  $i = 0$ 
4: while  $|Outliers| > tol \ \& \ i < M$  do
5:   for  $k$  in  $1 : k_{max}$  do
6:      $F_k = FuzzyCluster(k, D)$ 
7:      $V_k = ValidityIndices(F_k)$ 
8:   end for
9:   Select  $k$  using  $V$ 
10:   $i = i + 1$ 
11:   $FC_i = RefineClustering(F_k)$ 
12:   $Outliers = IdentifyOutliers(FC_i, D)$ 
13: end while

```

maximum number of clusters to consider at each repetition; tol is limit on the number of outliers that must be present for the algorithm to repeat; M is the maximum number of repetitions. There are four functions within the algorithm where choice is possible. These enable the tailoring of the algorithm to specific needs or implementations. The choices here can lead to the introduction of additional parameters.

2.1 Fuzzy Clustering

Fuzzy clustering is used to determine the grouping of students within a loop. The choice of fuzzy, as opposed to crisp, is because it provides a membership value for each student in each cluster. This is relied upon to determine outliers, S2.4. In this implementation fuzzy k-means [10] is used, although it would be possible to use any other fuzzy clustering algorithm in its place [14]. An effect of using fuzzy clustering in our approach is that the algorithm produces crisp divisions between loops and fuzzy divisions within.

2.2 Validity Indices

Validity indices provide a quantitative measure of cluster validation. Their calculation is a fundamental part of the clustering process and provides guidance when deciding on k , the number of clusters. There is a huge range of cluster validity indices [2] with a large subset focused on fuzzy clustering [26]. In this implementation, we use the six available in the FClust R package [12]. These include the Silhouette index [19], Fuzzy silhouette index [5], Partition coefficient [3], Modified partition coefficient [8], Partition entropy [4], and Xie and Beni index [27]. For each clustering solution, we record the value of k recommended by each validity index. The final value of k is the mode of these recommendations. In the case of two possible values for k , we chose the smallest.

2.3 Refining Clusters

Refining the clustering solution is an optional step that enhances the compactness of the final clusters on each loop. Given a solution, outliers once identified are removed from the data. The clustering procedure is then rerun with the same value of k to derive a tighter clustering solution that better represents that data and students that remain. This process repeats as required until a stable solution emerges and no outliers are present.

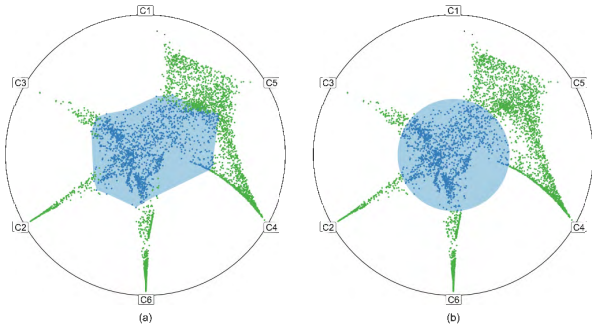


Figure 2: Radviz of membership for a solution with $k = 6$. Outliers are shown in blue. (a) Outliers as identified by (1). (b) Outliers as identified by (2).

2.4 Identifying Outliers

Identifying outliers is the most crucial step in the RFC algorithm. This process places the split in the data for each loop of the algorithm. Many strategies are possible with the choice depending on the application and chosen clustering procedure. [21], [15] and [13] all explore identifying outliers as part of the k-means clustering process. This is done for several reasons including creating more compact clusters. These process generally rely on distance measures to identify the outliers in the data.

The method presented here identifies outliers using the membership values from fuzzy clustering with two versions considered below. The rationale behind both of these approaches is that they seek a solution which places observations mostly within one or two clusters. Any observation split among three or more clusters is an outlier. For an instructor having a student predominately within only one or two clusters should help simplify the task of interpreting their behavior.

The first and simplest version of identifying outliers uses the maximum membership value m_i and the sum of the two highest membership values s_i for an individual observation i . The condition classifies an observation as an outlier if the values of m_i or s_i fall below some limit. Equation (1) places a limit of $\frac{1}{2}$ on the value of m_i and a limit on s_i that increases slowly from $\frac{1}{2}$ with increasing number of clusters k .

$$Outliers = \left\{ i \mid m_i < \frac{1}{2} \vee s_i < \frac{1}{2} + \frac{1}{k} \right\} \quad (1)$$

This condition will not work for $k = 2$ as s_i will always equal 1, and the condition on m_i will never be satisfied. In this case, one possible solution would be to place a stricter limit on m_i and drop condition on s_i .

The second approach makes use of the Radviz method of visualizing fuzzy cluster membership [18]. Radviz represents each cluster by a dimensional anchor and distributes each dimensional anchor evenly on a unit circle. Each observation corresponds to a point. The visualization connects each point to each anchor by a spring whose stiffness corresponds

to that observation's cluster membership for the associated anchor. The position is where the spring's tension is at its minimum. Imagine each anchor pulls on a data point with a strength equal to the cluster membership. The higher the membership value, the stronger the pull and the closer the data point to that anchor. The ordering of the anchors is essential, and work has been completed to determine the optimum position [9].

The advantage of this method is that it makes observations which are evenly split among multiple clusters evident as these will be close to the center of the visualization since they get equally pulled in all directions. Observations that are a member of a small number of clusters will generally be further from the center. An example of a Radviz, created using [1], from one stage of implementing the RFC algorithm with six clusters, can be seen in Figure 2. In part (a), outliers, as defined by (1) with $k = 6$, are colored in blue and are visible in the center of the graph.

An alternative to (1) is to use the position of each observation on the Radviz graph. Here outliers are defined as being those at the center of the graph within some circle of radius r and where x_i and y_i are the Cartesian coordinates of the position of the observation i in the visualization. The parameter r has a similar role to m in the fuzzy k-means algorithm in that it controls the fuzziness of the clusters. The larger r , the crisper the clustering.

$$Outliers = \{ i \mid x_i^2 + y_i^2 < r^2 \} \quad (2)$$

This method has the advantage of also working without modification for the case where $k = 2$, as points become spaced along a straight line. In this case, the condition 2 reduces to $m < r + \frac{1}{2}$. Figure 2(b) displays the outliers as identified by (2) using $r = 0.4$. We can see a significant overlap of points using both conditions.

3. EXPERIMENTAL RESULTS

3.1 Dataset

The data used to test the algorithm is from UCF's use of the Realizeit platform. The data encompasses $N = 5044$ students across 51 online and blended course deliveries across nine terms from 2015 to 2018. Both spring and fall terms last 15 weeks and the summer term is 12 weeks. The courses cover a range of disciplines including Psychology, Spanish, College Algebra, various Computing courses, and Nursing. UCF uses the platform in a variety of different contexts and the student learning in the platform contributes a more significant element of their final grade in some course than others. The data only contains first-time students; repeat students are filtered out.

3.2 Features

It is possible to define a distance metric for the progress trajectories in their raw form and to use the RFC algorithm. However, we can obtain more easily interpretable results for an instructor by extracting features from the trajectories that capture the key behavioral aspects. Through testing and iteration the following six were selected:

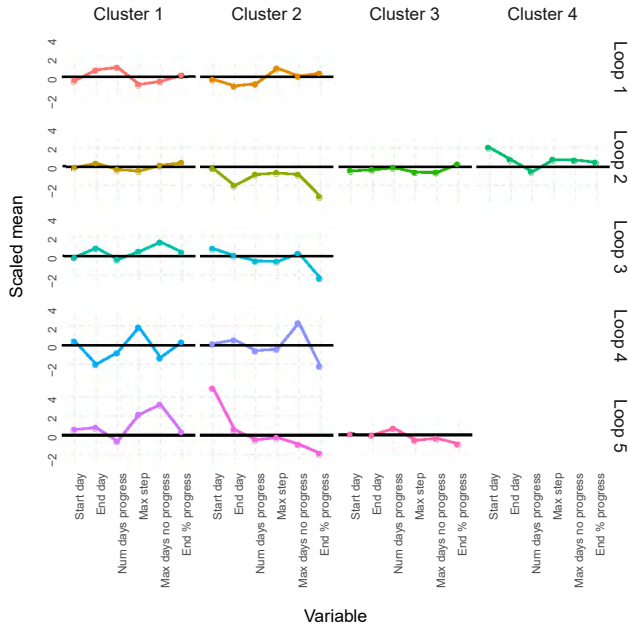


Figure 3: The center (mean) of each cluster on each of the key features for the normalized UCF data.

- **Start day** - The first day on which the student made progress.
- **End day** - The last day on which the student made some progress. The day on which the student reached their final % progress value.
- **End % progress** - The percentage of concepts mastered by the student by the end of the course.
- **Num days progress** - The number of days on which the student made progress.
- **Max step** - The single largest jump in progress on a single day.
- **Max days no progress** - Between the start and end day, the largest number of consecutive days on which the student made no progress.

One weakness of the chosen features is that they are only observable after the course is finished making an early prediction of behaviors difficult. Note that the trajectories do not capture all activities completed by the student, just those that increase their progress. For example, practices, revisions, or assessments are not evident in this data.

3.3 Clustering

The RFC algorithm made use of the fuzzy k-means algorithm with the fuzzy parameter set at $m = 2$. Note that the data was normalized before using fuzzy clustering. We set $k_{max} = 10$, $M = 10$, and $tol = 0.05N \approx 250$. The algorithm completed 5 loops and automatically produced 13 clusters in total. The breakdown of clusters per loop and the weight of each cluster is provided in Table 1. We use weight since a student belongs only partially to any one cluster.

Table 1: Cluster and Loop weights W , % and deviation from average δ .

Cluster	W	%	δ	W	Total %	$\bar{\delta}$
C1L1	1685.6	33.42	0.60	3244	64.31	0.57
C2L1	1558.4	30.90	0.55			
C1L2	280.6	5.56	0.29			
C2L2	125.3	2.48	1.25			
C3L2	223.7	4.44	0.38			
C4L2	285.3	5.66	0.87	915	18.14	0.70
C1L3	202.0	4.00	0.61			
C2L3	91.0	1.80	0.75			
C1L4	129.4	2.57	1.12	340	6.74	1.08
C2L4	210.6	4.18	1.04			
C1L5	63.3	1.25	1.29	252	5.00	1.07
C2L5	76.9	1.52	1.52			
C3L5	111.8	2.22	0.44			

The first loop captures the standard approach of applying the fuzzy k-means algorithm once and stopping (if the refinement step is excluded). It is the clusters on loop two to five that are new, and it is here that we find the outlier behaviors that would be missed by the standard approach. Notice that the number of students clustered on each loop generally decreases as the loop count increases. Another point is that these “outliers” account for over 30% of the students.

Figure 3 visualizes the center (mean of the normalized data) of each cluster for each feature. Figure 4 displays the trajectories belonging to each cluster with a membership greater than 0.5. The students with the highest membership values for each cluster are shown in black, and these can be taken as prototypes for each cluster to help interpretation. Note that some of the trajectories in each cluster vary considerably from the prototypes due to the fuzzy nature of the clusters and likely have membership values close to 0.5. The noise present in the clusters on the final loop suggests that perhaps the algorithm stopped too early and allowing additional loops could uncover new behaviors.

From an examination of these graphs, we can see that some outlier behaviors are entirely different from the most common behaviors found on the first loop. There are certain similarities in some cases but enough of a difference to make them worthy of being categorized as separate behaviors.

The clusters found on loop one represent more successful behaviors in that the students generally finish over 50% of the concepts. The first that represents unsuccessful behavior appears on loop two, with more appearing on later loops. Below we provide notes on some of the individual behaviors. A detailed analysis is beyond the scope of this paper.

- Students in cluster 1 on loop 4 (C1L4) master all the concepts in a short period right at the start of the course.
- C2L5 are the students who generally did too little too late.
- C3L5 are students who start well but for some reason stopped with about a month to go.

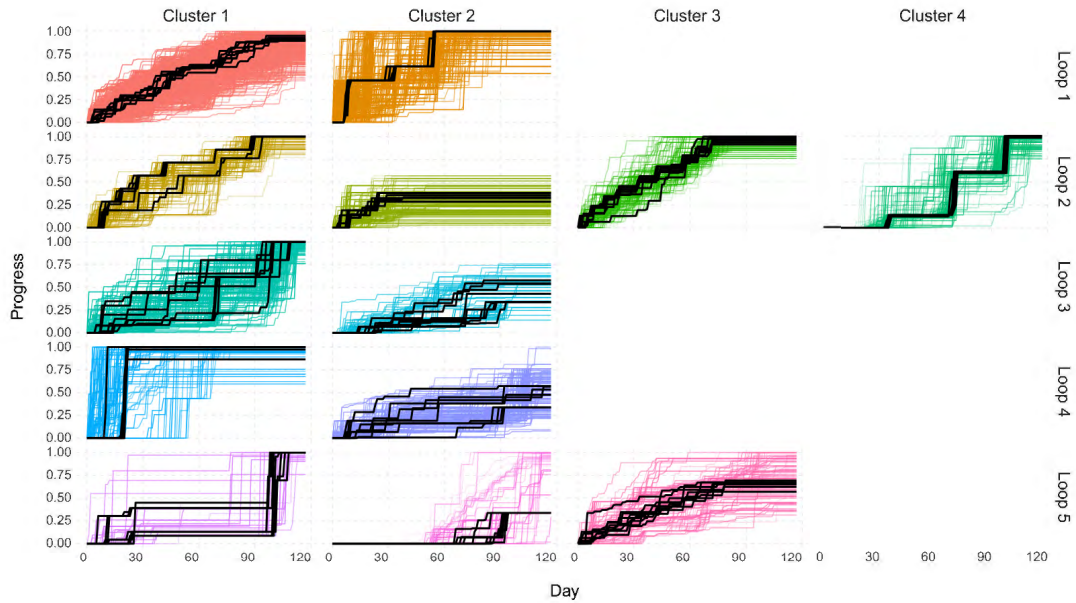


Figure 4: The trajectories belonging to each cluster in the UCF data. The most representative (highest membership values) members of each cluster are shown in black.

- C2L1 and C4L2 are similar in that they make their progress is a small number of large steps. The difference is when in the course that progress takes place.
- C1L5 are students who have a long dormant period in the middle of the course and leave everything to the last minute.

As expected the clusters found on the early loops tend to capture behaviors that are close to the “average,” whereas later loops have clusters that are more different. We demonstrate this by examining the cluster centers displayed in Figure 3. The deviation of a cluster center from the mean (solid black line) is an indication of how far the behavior is from the average. Table 1 provides this deviation, calculated as the mean absolute difference, for each cluster and loop. We see that in general later clusters capture more extreme behaviors. The cluster closest to the average is C1L2, but only represents about 5.5% of the students. The cluster furthest from the average is C2L5 and represents about 1.5% of students. What makes these students stand out is their late start time and low level of progress.

3.4 Comparison

To highlight the limitations of standard approaches, we applied both crisp and fuzzy k-means to the UCF dataset. In summary, these algorithms produce a much smaller number of clusters and do not capture the same range of outlier behavior as those captured by the RFC algorithm. Table 2 display the results from fuzzy k-means for various values of the fuzziness parameter m . For each value of m , the table provides the validity indices, the selected number of cluster k , and the number of outliers based on (2). The value of $m = 2$ is the default and corresponds to applying just one loop of the RFC without refinement. We observe that we get more clusters and fewer outliers as $m \rightarrow 1$. Indeed the validity indices suggest that $m = 1.01$ is the best solution

Table 2: Results of fuzzy k-means for various values of m including the validity indices and number of outliers.

m	k	SIL.F	SIL	PC	PE	MPC	XB	Out.
1.01	5	.58	.58	1.0	.00	1.0	.29	1
1.2	4	.59	.56	.94	.11	.92	.36	66
1.4	3	.58	.51	.83	.31	.75	.46	366
1.6	3	.60	.50	.73	.50	.59	.49	864
1.8	3	.62	.49	.63	.66	.44	.52	1438
2.0	2	.52	.44	.69	.48	.37	.54	1606

of those presented. However, with this solution, we only get five clusters, and these contain high levels of noise and are therefore can be challenging for instructors to interpret. This low number of clusters does not accurately capture the full range of behaviors apparent in the data.

With the solution improving as $m \rightarrow 1$ the logical step to take is to set $m = 1$ and perform simple crisp clustering using the k-means algorithm. We performed this using the NBClust R package [7] which provides a collection of 23 appropriate validity indices to help with the choice of k . Of these, 7 proposed 3 clusters, followed by 5 indices proposing 7 clusters. Both values lead to the same conclusion as we arrived at with fuzzy clustering; that is, the number of clusters does not adequately capture the full range of behaviors.

4. CONCLUSIONS AND FUTURE WORK

The RFC algorithm has allowed us to uncover outlier behaviors that are in some cases very different to the most common behaviors found on loop 1, and in other cases appear visually similar but represent a very different type of learning behavior. The behavioral clusters found here are by no means an exhaustive list. Adjusting the parameters of the algorithm, for example, by changing the parameters

that control the fuzziness of the clustering, would possibly allow more outlier behaviors to emerge.

The purpose of this paper was to describe and demonstrate the RFC algorithm in its current form. Many possible improvements and extensions could be carried out. Once the RFC algorithm has finished, it is possible that clusters on a later loop could better capture a student that belongs to some clusters on an earlier loop. One extension could be to carry out a refinement process moving students from earlier to later clusters. Potentially we can achieve further improvements by including additional features that capture other aspects of behaviors or by applying a weighting to features that are considered more critical.

Lakoff [20] puts the clustering process this way, “Categorization is not to be taken lightly. There is nothing more basic than categorization to our thought, perception, action and speech” ([20] pg. 5). Identifying these student trajectories as either subordinate, superordinate of basic level create a substantial educational responsibility in the adaptive learning environment where students have control time, pace and feedback. If John Carroll [6] was correct in that learning is a function of time spent and time needed then the question is what resources do various student cohorts require. We argue that the clustering process can help in the better understanding of what it will take to help larger numbers of students become successful. As we explore these procedures several questions emerge. If and when will the process become excessively granular and dysfunctional how can these processes be integrated into the educational environment? Can these methods contribute to resolving achievement inequality? Finally, the question remains about whether the clusters exhibit a categorical structure with meaningful prototypes that respond to instructional interventions.

5. REFERENCES

- [1] Y. Abraham. *Radviz: Project Multidimensional Data in 2D Space*, 2016. R package version 0.7.0.
- [2] O. Arbelaitz, I. Gurrutxaga, J. M. Pérez, and I. nigo Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013.
- [3] J. Bezdek. Cluster validity with fuzzy sets. *Journal of Cybernetics*, 3:58–73, 1974.
- [4] J. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.
- [5] R. Campello and E. Hruschka. A fuzzy extension of the silhouette width criterion for cluster analysis. *Fuzzy Sets and Systems*, 157:2858–2875, 2006.
- [6] J. B. Carroll. A model of school learning. *Teachers College Record*, 64:723–733, 1963.
- [7] M. Charrad, N. Ghazzali, V. Boiteau, and A. Niknafs. NbClust: An R package for determining the relevant number of clusters in a data set. *Journal of Statistical Software*, 61(6):1–36, 2014.
- [8] R. Dave. Validating fuzzy partitions obtained through c-shells clustering. *Pattern Recognition Letters*, 17:613–623, 1996.
- [9] L. Di Caro, V. Frias-Martinez, and E. Frias-Martinez. Analyzing the role of dimension arrangement for data visualization in radviz. In *Advances in Knowledge Discovery and Data Mining*, pages 125–132. Springer, 2010.
- [10] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973.
- [11] C. Dziuban, C. Howlin, J. Constance, and P. Moskal. An adaptive learning partnership. *Educause Review*, Dec. 2017.
- [12] M. Ferraro and P. Giordani. A toolbox for fuzzy clustering using the r programming language. *Fuzzy Sets and Systems*, 279:1–16, 2015.
- [13] G. Gan and M. K.-P. Ng. k-means clustering with outlier removal. *Pattern Recognition Letters*, 90:8–14, 2017.
- [14] A. Gosain and S. Dahiya. Performance analysis of various fuzzy clustering algorithms: A review. *Procedia Computer Science*, 79:100–111, 2016.
- [15] V. Hautamäki, S. Cherednichenko, I. Kärkkäinen, T. Kinnunen, and P. Fränti. Improving k-means by outlier removal. In H. Kalviainen, J. Parkkinen, and A. Kaarna, editors, *Image Analysis*, pages 978–987. Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
- [16] K. He, Y. Li, S. Soundarajan, and J. E. Hopcroft. Hidden community detection in social networks. *CEUR Workshop Proceedings*, 1828:89–93, Feb. 2017.
- [17] K. He, S. Soundarajan, X. Cao, J. Hopcroft, and M. Huang. Revealing multiple layers of hidden community structure in networks. <http://arxiv.org/abs/1501.05700v1>, 2015.
- [18] P. Hoffman, G. G. Grinstein, and D. Pinkney. Dimensional anchors: A graphic primitive for multidimensional multivariate information visualizations. In *Workshop on New Paradigms in Information Visualization and Manipulation*, pages 9–16, 1999.
- [19] L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, 1990.
- [20] G. Lakoff. *Women, fire, and dangerous things: What categories reveal about the mind*. Chicago: University of Chicago Press, 1987.
- [21] H. Liu, J. Li, Y. Wu, and Y. Fu. Clustering with outlier removal. <http://arxiv.org/abs/1801.01899>, 2018.
- [22] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33(1):135–146, 2007.
- [23] E. Rosch. Natural categories. *Cognitive Psychology*, 4:328–350, 1973.
- [24] E. Rosch. Cognitive reference points. *Cognitive Psychology*, 7:532–557, 1975.
- [25] A. Vellido, F. Castro, and A. Nebot. *Clustering Educational Data*. In *Handbook of Educational Data Mining*, pages 72–92. CRC Press, first edition, 2011.
- [26] W. Wang and Y. Zhang. On fuzzy cluster validity indices. *Fuzzy Sets and Systems*, 158(19):2095–2117, 2007.
- [27] X. Xie and G. Beni. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:841–847, 1991.

Content-based Course Recommender System for Liberal Arts Education

Raphaël Morsomme
University College Maastricht
Zwingelput 4
6211 KH Maastricht

raphael.morsomme@maastrichtuniversity.nl

Sofia Vazquez Alferez
University College Maastricht
Zwingelput 4
6211 KH Maastricht

sofia.vazquezalferez@maastrichtuniversity.nl

ABSTRACT

Liberal Arts programs are often characterized by their open curriculum. Yet, the abundance of courses available and the highly personalized curriculum are often overwhelming for students who must select courses relevant to their academic interests and suitable to their academic background. This paper presents the course recommender system that we have developed for the Liberal Arts bachelor of the University College Maastricht, the Netherlands. It aims to complement academic advising and help students make better-informed course selections. The system recommends courses whose content best matches the student's academic interests, issues warnings for courses that are too advanced given the student's academic background and, in the latter case, suggests suitable preparatory courses. We base the course recommendations on a topic model fitted on course descriptions, and the warnings on a sparse predictive model for grade based on students' past academic performance and level of academic expertise. Preparatory courses consist of courses whose content has the best preparatory value according to the predictive model. We find that course recommendations are relevant for a wide range of academic interests present in the student population and that students found recommendations for courses at other departments especially helpful. The preparatory courses often lack coherence with the target course and need to be improved.

Keywords

Education, recommender system, warning, topic model, grade prediction.

1. INTRODUCTION

The Bachelor in Liberal Arts offered at the University College Maastricht, the Netherlands, is an honors program characterized by an open curriculum. The program allows students to design their curriculum in a fairly free fashion: more than 75% of the educational credits are free, the college offers over 150 courses covering a wide range of topics from artificial intelligence, to conflict resolution and to pop songs, and students can take up to one year's worth of courses outside of the college. This freedom allows students to tailor their curriculum to their own interests; but the abundance of courses available makes the selection of courses overwhelming. First, the number of courses offered at the

12 departments of the university is too large for students to have an overview of which ones match their academic interests. Second, since each liberal arts student has a unique curriculum, it can be difficult for them to determine if they have covered the necessary prerequisites for a particular course or if the course's level is too advanced given their academic background. A recommender system that suggests courses whose content matches students' academic interests, issues a warning for courses too advanced and, in the latter case, provides suitable preparatory courses would therefore be extremely beneficial. Not only would it increase the students' information position, thereby improving self-advising, but it would also improve academic advising when used as an agenda-setting tool.

Our course recommender system achieves these three goals: course suggestion, warning issuance and preparatory course advice. To receive course suggestions, the student enters her/his academic interests into the system which returns the 20 courses whose content best matches them. In practice, the student selects key words from a predetermined list that represent her/his academic interests. The course recommender system then uses a topic model to identify the courses whose content best matches the topics corresponding to the selected key words (see Figure 1). To receive warnings, students provide their transcript and indicate which courses they are considering for the following term. The system issues a warning for courses that it identifies as too advanced given the student's academic background. In practice, the student enters her/his student ID with which the system extracts her/his past academic performance and the expertise that she/he has acquired in various topics. From these, the system uses a predictive model to estimate the grade that the student will obtain in the selected courses and issues a warning when the predicted grade is a fail (see Figure 2). Each warning issued is then accompanied by a list of preparatory courses whose content has the best preparatory value according to the predictive model.

2. RELATED WORK

Identifying courses that are both of interest to the students and of an appropriate level is a task that has recently gained attention in the literature. Gulzar et al. [8] propose a recommender system that uses information retrieval techniques to select courses based on students' interests. Their system uses key words to search the space of possible courses and tries to improve the quality of the query by finding synonyms and generating N-grams so that the search returns a higher number of courses. Then, they use an ontological model to expand the search even further and retrieve courses related to the previously extracted courses in the ontological model. In this context, an ontological model is a knowledge model that represents relationships between concepts of a previously specified domain, such as 'Computer Science' [7]. This system is content-based since it is the contents of the courses

Raphael Morsomme and Sofia Vazquez Alferez "Content-based Course Recommender System for Liberal Arts Education" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 748 - 753

that are matched to the concepts of the ontological model or the key words of the query. In this manner, the recommender system allows the interest of the students to be matched to the contents of the course. However, the system suffers from several drawbacks: first, the domains (e.g. Computer Science or Medicine) from which the ontological models are built must be defined a-priori [7]. Second, the recommender system is dependent on a well-built database that is not always available at interested institutions.

Code	Course	Because you selected
unknown	Arts and Culture: Policy and Politics	art, language, cultural
HUM2013	The Presence of Art: Reintegrating Modern and Contemporary Art	art, language, cultural
unknown	Globalization Seminar & Symposium	art, cultural, language
unknown	Museum Meanings	art, language, cultural
HUM1011	Introduction to Art: Representations, Performances and Interactions	art, cultural, performance
unknown	Paper Minor Arts and Heritage	art, language, cultural
HUM2031	Cultural Studies II: Visual Cultures	art, cultural, performance
unknown	World War II and Memory	language, art, cultural
unknown	Modernity and Its Discontents	art, language, cultural
HUM2056	Cultural Remembrances	art, cultural, performance
unknown	Great Books and Debates	language, art, cultural
HUM3042	Biomedicine: An Evolutionary Approach to Art, Literature, Music and Religion	art, cultural, performance
unknown	Kunst- en cultuurbeld	language, art, cultural
unknown	The Birth of Reason	language, art, cultural
HUM2043	Film Art	art, performance
HUM2047	The Future of Literature?	art, cultural, language
unknown	Placing Europe: Cities, Regions, Borders	language, cultural, art
HUM2022	Digital Media	art, cultural, performance
unknown	Practicalities of Policy Making	language, art, cultural

Figure 1. Course suggestions.

Course	Warning	Preparatory Courses
SC2040	Red	XXXX000 - Statistics I XXXX000 - Cognitive Enhancement SC20035 - Biochemistry CHE2004 - Biochemistry SC20061 - Statistics I
SC2050	Orange	CHE1001 - Introduction to Natural Sciences Chemistry SC1004 - Introduction to Chemistry XXXX000 - Child Neuropsychology SC30000 - Adult Neuropsychology: An Introduction SC10008 - Introduction to Academic Skills
HUM3034	Orange	XXXX000 - Interpersonal Neuroscience: Evaluating claims about food, the XXXX000 - Social Neuroscience XXXX000 - Action XXXX000 - Cognitive Enhancement XXXX000 - Introduction to Natural Sciences: Mathematical Foundations of Physics

Figure 2. Warnings and preparatory courses.

Bydžovská [3] develops a recommender system that takes into account a student's past performance and interest profile to make course recommendations. Students' interests are defined in a narrow sense, that is, a course is considered of interest if a student has taken the course or marked it as a favorite in the university system. Course recommendations based on interest are then issued via a collaborative filtering approach: the suggested courses are the courses most selected by other students in the same field of study, or those that were taken by the n-most similar students that already graduated. To detect risk of failure, Bydžovská [3] predicts grades using classification and regression, or nearest neighbor, depending on the course. Warnings are issued after binning the predicted grades into excellent, good, or bad. The main innovation of the system is that it proceeds to include social behavior and take into account courses taught by a favorite teacher or taken by similar students. Although the system attempts to handle both interest and appropriateness of a course's level, it suffers from three major disadvantages: first, it does not provide the kind of transparent recommendation that would allow students to reflect on their course selection because the content of the course is not explicitly taken into account. Second, it does not give students suggestions on how to address their deficiencies.

Third, it does not permit students to change their interest, which is particularly important in a liberal arts context where students go through a broad exploratory phase before specializing.

Bakhshinategh et al. [1] address the issue of recommending courses that help students overcome their deficiencies whilst accounting for changes over time. They view a study program as a path to obtain graduating attributes (skills, qualities, understandings) and rank the impact that each course has on promoting those graduating attributes for a student who took the course. The ranking is done through self-assessment by students after completing the course. The recommender system then uses collaborative filtering to find courses that score highest on promoting a targeted graduating attribute for a student who wishes to develop it further. Thus, if a student lacks "analytical skills", the system identifies courses that improve these skills so that a student comes closer to the level of "analytical skills" that is required for graduation. This system can be used to find preparatory courses for other courses by shifting from graduating attributes to attributes required to succeed in a course. The main disadvantage is that the impact of each course is found through self-assessment rather than in a data-driven way.

Jiang et al. [11] take a different approach to find preparatory courses by using recurrent neural networks to develop a goal-based course recommender. A student specifies a course that they wish to take, along with the grade that they desire to achieve, and the system uses their transcript to find personalized preparatory courses. Although this approach finds preparatory courses in a data-driven way, it does so at the expense of transparency, which makes a student's reflective decision-making process more difficult and provides no direct insight for academic advising.

We use a topic model to extend Bydžovská's [3] use of students' interest. This provides a more flexible and realistic interpretation of a student's interests and how they change over time. Moreover, we use a topic model to expand the search of relevant courses in the manner that by Gulzar et al. [8] use ontological models. The advantage of a topic model is that topics are learned from the data and must not be known in advance. Our system also supplements recommendations with explanations and additional information to help students make well-informed course selections.

3. DATA

We use two types of data: student data and course data.

The student data consists of anonymized course enrollment information. We use the transcripts of the 2,526 students of the liberal arts program between 2008 and 2019 with a total of 79,245 course enrollments. We exclude enrollments with a missing grade which indicates that the student either dropped the course or fail the attendance requirement. In the latter case, the data set contains an observation corresponding to the resit. Table 1 presents the student data. Each row contains an anonymized student ID, a course ID, a year and semester, and the obtained grade.

The course data consists of the 2018-2019 course catalogues of 5 departments of Maastricht University: European Studies, University College Maastricht, University College Venlo, Psychology and Science Program. These catalogues contain a one-page description of 490 courses. Table 2 presents the textual data in the tidy format with one row per document-term [18]. We process the data following common cleaning procedures [13]: we tokenize the individual terms, stem them with the Hunspell dictionary and remove common stop words, numbers between 1 and 1,000, and terms occurring less than 3 times in the data set.

Table 1. Example of student data

Student ID	Course ID	Academic Year	Period	Grade
44940	CAP3000	2009-2010	4	8.8
37490	SSC2037	2009-2010	4	8.4
71216	HUM1003	2010-2011	4	6.8
44212	SSC2049	2010-2011	2	8.4
85930	SSC2043	2011-2012	1	4.3
14492	COR1004	2012-2013	2	8.5
34750	HUM2049	2013-2014	5	6.0
32316	SSC1001	2013-2014	1	8.5
22092	SCI1009	2014-2015	1	6.4
19512	COR1004	2016-2017	5	7.0

Table 2. Example of course data

Course ID	Course Title	Department	word
HUM3034	World History	UCM	understand
HUM3034	World History	UCM	major
HUM3034	World History	UCM	issue
HUM3034	World History	UCM	episode
HUM3034	World History	UCM	shape
HUM3034	World History	UCM	history
HUM3034	World History	UCM	mankind
HUM3034	World History	UCM	focus
HUM3034	World History	UCM	theme
HUM3034	World History	UCM	topic

4. METHODOLOGY

4.1 Overview

Figure 3 presents a diagram of the course recommender system. We start by fitting models to the data. We use the Latent Dirichlet Allocation statistical model to fit a topic model on the course data and the lasso penalty to fit a series of sparse multiple linear regression models for grade prediction to the student data. A model is fitted to each course. Their inputs consist of the students' past academic performance and level of expertise in the topics which we derive from their transcript with the topic model.

These models generate intermediate results from the user's input. We use the topic model to infer the student's academic interests from the key words that she/he has entered into the system and the regression models to predict the grades that the student will obtain in the course she/he selected based on her/his transcript.

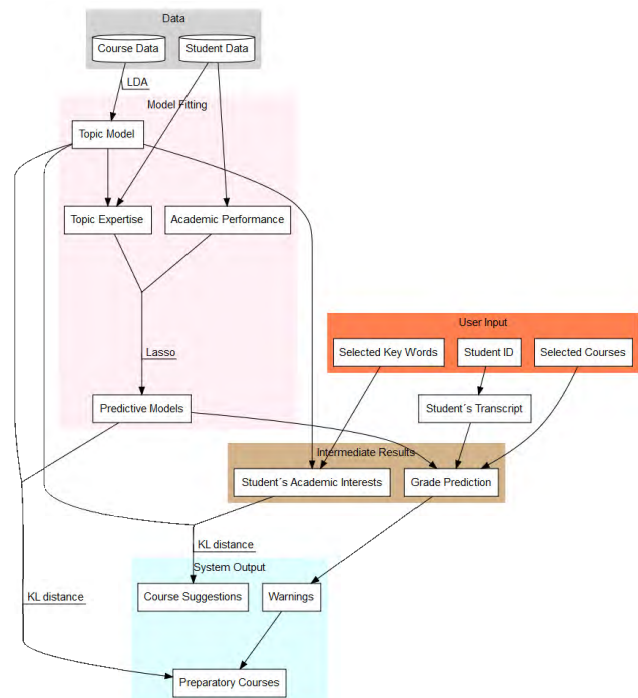
The system's outputs are based on these intermediate results. The course suggestions consist of the 20 courses whose content best matches the student's academic interest in terms of Kullback-Leibler distance. Warnings are issued when the predicted grade is a fail. For each warning issued, we indicate the 5 courses whose content has the best preparatory value according to the regression model. The preparatory value of a course is estimated with the Kullback-Leibler distance of its topic distribution to the coefficient estimates of the topic variables in the linear regression.

All computations are realized on the environment for statistical computing and graphics *R* [16, 13, 10, 4, 19, 20].

4.2 Topic Model

We use the Latent Dirichlet Allocation (LDA) generative probabilistic model and the Gibbs sampling algorithm to fit a topic model to the course data.

The LDA model conceptualizes topics as a probability distribution over a finite set of words (in this case, the vocabulary of the course data), and a document (i.e. a course description) as a sequence of N words, where each word was generated by drawing from a probability distribution over topics specific to that document [2]. Thus, each word belongs to all topics but with different probabilities, and all topics are present in each course but with different weights. Figure 4 and Figure 5 respectively show the word distribution in two topics and the topic distribution in a course based estimated by the topic model fitted on the course data. Technically, the LDA model generates a document as follows. First, the word distribution β for each topic is determined by $\beta \sim \text{Dirichlet}(\delta)$ and the topic weights θ for each document are determined by $\theta \sim \text{Dirichlet}(\alpha)$. Second, each of the N words of the document is chosen by choosing a topic $z \sim \text{Multinomial}(\theta)$ and then choosing a word from a multinomial probability distribution conditioned on the topic z .

**Figure 3. Diagram of the course recommender system.**

Gibbs sampling is a Monte Carlo Markov Chain (MCMC) technique for successively sampling conditional distributions of variables whose distribution over states converges to the true distribution in the long run [5]. Gibbs sampling generates posterior samples by sweeping through each variable and sampling from its conditional distribution when the other variables are fixed to their current values. Phan et al. [14] used Gibbs sampling to learn the distributions β and θ for the LDA model. In this case, δ and α are the prior distributions for Gibbs sampling, acting as hyper-parameters that respectively determine how sparse the distributions of words in topics and topics in documents are. Gibbs sampling picks each word in the vocabulary and estimates the probability of assigning the current word to each topic conditioned on the topic assignments of all other words. With this conditional distribution, given a document, a topic is sampled and assigned as the new topic assignment for the current word. Then, with the distribution of words per topic, we compute the conditional probability of the topics given an observed

document. Since Gibbs sampling is a MCMC, the distribution sampled from a large number of iterations approximates the target distribution [5], enabling us to infer β and θ .

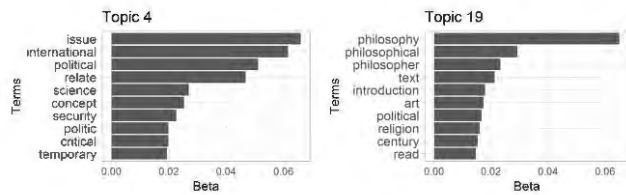


Figure 4. Term distribution in two topics. Topic 4 corresponds to international politics and topic 19 to philosophy.

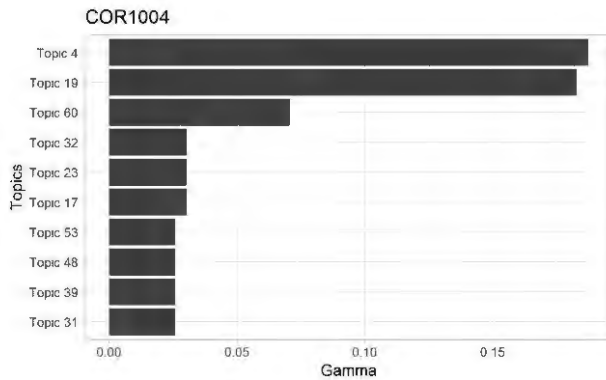


Figure 5. Topic distribution in the core course COR1004 Political Philosophy. The course is characterized by topic 4 (international politics) and topic 19 (philosophy).

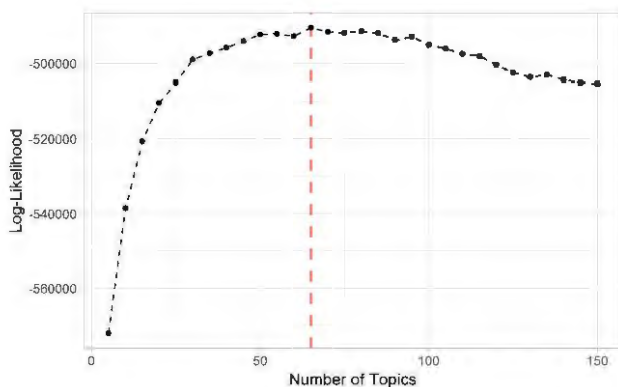


Figure 6. Model selection based on log-likelihood. The maximum-likelihood model has 65 topics.

4.2.1 Model Selection

This procedure requires that we fix *a-priori* the number of topics (k) to be inferred. We trained 30 models with $k = 5, 10, \dots, 150$. We set α to $50/k$ and δ to 0.1 as suggested by Griffiths & Steyvers [6]. For Gibbs sampling, we run 6,000 iterations with a burn in of 1,000 iterations and sample every 100 iterations. To avoid being stuck in a local optimum, we use 10 random initializations to explore the model space and keep the best model with respect to the log-likelihood. We then select the number of topics yielding the model with the largest log-likelihood [6]. Figure 6 shows the log-likelihood of the topic models that we trained. The model with 65 topics has the maximum log likelihood. In order to increase the quality of the selected model, we refit it with more iterations (16,000 iterations with a burn in of 2,000 iterations and 20 random starts; the other parameters are kept the same).

4.3 Warnings

We fit a sparse multiple linear regression model for grade prediction to each of the 132 courses currently offered at the college that have had more than 20 student enrollments since 2008. We regularize the models with the lasso penalty [17]. The set of predictors consists of students' past academic performance and their level of topic expertise at the start of the course. Students' past academic performance consists of 6 variables corresponding to their general and concentration-specific GPA's (humanities, natural sciences, social sciences, skills and projects). Students' topic expertise consists of a set of 65 variables (one per topic of the topic model) which indicate how much knowledge the student has acquired about the topic during her/his studies. A topic expertise variable corresponds to the sum of the topic's importance in the courses taken by the student (as estimated by the topic model) weighted by the grades. We assume that students who obtain 10/10 acquired all the topic-related knowledge present in the course while those obtaining 5/10 acquired half of it. Tables 3a, 3b and 3c and Figure 7 show a toy example of the contribution of individual courses towards a student's topic expertise.

Since the number of predictors is large, we regularize the models with the lasso penalty to increase their accuracy. The lasso penalty shrinks the coefficient estimates of the model, thereby reducing its variance. For each model, we use 10-fold cross-validation (CV) to find the lasso tuning parameter λ that minimizes the CV mean absolute error, a more robust loss function than the squared error [9]. Figure 8 presents the distribution of the CV mean absolute error for the 132 prediction models. The model for the course *PRO2004 Academic Debate* has the smallest prediction error (0.38 grade point) and the model for *SCI3006 Mathematical Modelling* the largest (1.80 grade point). The mean CV error weighted by the number of students enrolled in the course is 0.78, the median is 0.78 and the standard deviation is 0.28.

To receive a warning, the user enters into the system her/his student ID and the list of courses that she/he is considering for the coming term. The system uses the student ID to extract her/his transcript, from which her/his past academic performance and topic expertise are determined. We then use the regression models to predict the grades that the student will obtain in the selected courses and issue a warning for the fail grades (see Figure 2).

4.3.1 Rule-based Warnings

We initially explored an alternative approach for warnings based on association rules. We used the SPADE algorithm [21] to identify sequences in the students transcripts of the type $\langle \text{fail course A} \rangle \Rightarrow \langle \text{fail course B} \rangle$ or $\langle \text{not take course A} \rangle \Rightarrow \langle \text{fail course B} \rangle$ and considered sequences with a support superior to 10 students, a confidence superior to 0.4 and a lift superior to 1.1. Warnings were issued when a student selected a course for which one of the selected rules predicted a failure.

The transparency of this approach motivated its initial adoption; but it turned out to be unsuitable to our case. First, given the small size of the student data and the fact that relatively few students fail courses at the college, only 21 rules met the criteria. Second, this approach ignores the fact that skills necessary to perform well in a particular course can be acquired by taking a *combination* of courses. To tackle the first issue, we considered a relaxed version of the rules that substitutes a $\langle \text{fail course A} \rangle$ with a $\langle \text{obtain less than 6.5 in course A} \rangle$. The number of rules meeting the relaxed criteria increased to 185. Yet, the second issue remained and led us to consider regression models that use topic expertise as a proxy for the skills necessary to perform well in a course.

Table 3a. Toy example: topic distribution in 3 courses

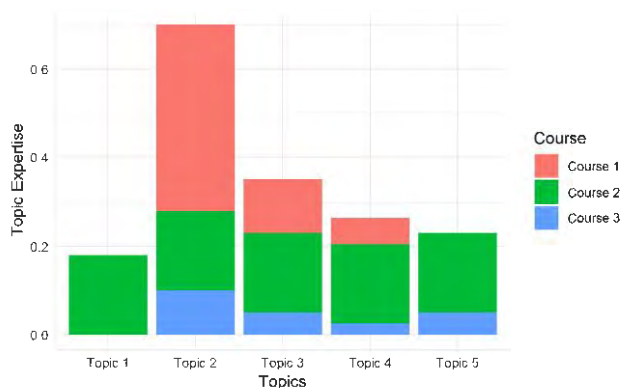
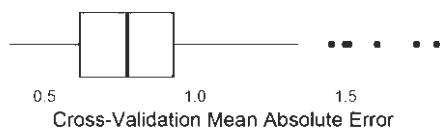
Course	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
Course 1	0.0	0.7	0.2	0.1	0.0
Course 2	0.2	0.2	0.2	0.2	0.2
Course 3	0.0	0.4	0.2	0.1	0.2

Table 3b. Toy example: transcript

Course	Grade
Course 1	6/10
Course 2	9/10
Course 3	2.5/10

Table 3c. Toy example: course contribution to topic expertise

Course	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
Course 1	0.00	0.42	0.12	0.060	0.00
Course 2	0.18	0.18	0.18	0.180	0.18
Course 3	0.00	0.10	0.05	0.025	0.05

**Figure 7. Toy example: course contribution to a student's topic expertise. We use these variables to predict grade.****Figure 8. Distribution of the cross-validation mean absolute error in the 148 predictive models.**

4.4 Course Recommendation

To provide course recommendations, we identify courses whose content best matches the academic interests of the students. We use the Kullback-Leibler distance, an asymmetric measure of the difference between two probability distributions [12], to estimate the degree to which a course's topic distribution (as estimated in the topic model) corresponds to the normalized academic interests of the student. The system returns the 20 courses with the smallest such distance. A student's academic interests profile consists of a numeric vector indicating the interest of the student in each of the topics. It corresponds to the sum of the selected key words' contribution to the topics in the topic model. In order to assist students in selecting key words, we preselect the 10 terms most relevant to their topic expertise profile (as defined in section 4.3). To make the system as informative and transparent as possible, each recommendation includes the three selected key words with the most relevance to the course. Here, a term's *relevance*

corresponds to the sum of the term's contribution across the topics weighted by the importance of the topics in the student's academic interests profile or student's topic expertise profile.

4.5 Preparatory Courses

In order to help students plan their curriculum, each warning is accompanied by a list of suitable preparatory courses. Similarly to the regression models built for the warnings, we fit a lasso-regularized multiple linear regression model for grade prediction to each course; but this time, the input only consists of the students' topic expertise. A positive coefficient estimate indicates that more knowledge of the topic is associated with larger grade in the course. For each course, the preparatory courses consist of the 5 courses (excluding advanced courses) whose topic distribution has the smallest KL distance to the course's regression's normalized coefficient estimates.

5. RESULTS

We used expert validation to evaluate the system's usefulness: current students, alumni and members of academic advising interacted with the system and commented on their experience.

We find that the system recommends course that are potentially useful to the students, thereby helping them make better-informed course selections. First, students value the system's ability to consider multiple interpretations of the same term e.g. the term *function* in mathematics and in biology. This feature of the system stems from the possibility for a term to have a large weight in several topics. Second, many users were surprised that they do not need to enter key words present in the course description for the course to be recommended. Since topics act as a buffer between the key words entered in the system and the course descriptions, students merely need to choose key words that characterize topics present in the course. This allows them to focus on their academic interests when selecting key words as opposed to thinking about the courses that might interest them. Third, they found that self-selected key words yield recommendations that are more useful than those stemming from the preselected key words. This pattern is due to the presence of topics related not to the content but the structure of the courses. For instance, topic 25 is dominated by the terms *paper*, *write* and *assessment*. A student's topic expertise profile therefore contains topics related to the *structure* of the courses that they have taken, which, for a student focusing on film art, leads to the preselection of the terms *research*, *method*, *period*, and *skill* along with *film*, *gender*, *literature* and *culture*, and the recommendation of the courses *Research Methods: Interviewing*, *Research skills*, and *Research Project*. Excluding structured-related key words solves the issue and results in suggestions of potentially interesting courses: *Narrative Media*, *Pop songs and poetry*, and *Cultural Studies II*. We therefore include an *opt-out* option to cancel key word preselection. Fourth, students found recommendations for courses at other departments particularly useful: in most cases, they ignored that these courses existed or that their content matched their academic interests.

Students wished that warnings also included low grades. We therefore provide *red* warnings for predicted fail grades ($< 5.5/10$) and *orange* warnings for low ones (between $5.6/10$ and $6.5/10$).

Users were enthusiastic about the preparatory courses; they found it very beneficial to receive suggestions of how to prepare for a particular course. Unfortunately, the preparatory courses returned by the system often lack coherence with the target course. For instance, the list of preparatory courses for the course *World*

History contains the course *Nutritional Neuroscience*. These incongruences may stem from the presence of structure-related topics in the topic model combined with the fact that the lasso penalty shrinks most coefficient estimates to 0. Hence, it is possible that the regression model for grade prediction of some courses only has non-zero coefficient estimates for structure-related topics, hence yielding preparatory courses characterized by these structure-related topics, and not the content-related topics.

6. FUTURE WORK

This course recommender system is a work in progress and the difficulties detailed above indicate three pathways for future work.

First, we need to differentiate structure-related and content-related topics. This seems particularly difficult to do. One approach is to manually inspect the topics that are most prevalent in the corpus of documents and reduce the weight of the structure-related ones.

Second, in order to increase the coherence between preparatory courses and target course, we could impose that their content must be related. The KL distance could be used to accomplish this. We could also take the personalized approach of Jiang et al. [11].

Third, since the topic model has a central place in the system, we plan to improve it by (i) expanding the course data to course manuals (20-page document offering a detailed description of a course's content) and the material covered in the course e.g. academic articles, textbook chapters, and (ii) using a structural topic model that uses covariates to build the model and calibrate topic prevalence and topic content depending on metadata [15] to take into account the origins (department) of the course data.

7. ACKNOWLEDGMENTS

Our thanks to the University College Maastricht, Maastricht University, the Institute of Data Science, and the Department of Data Science and Knowledge Engineering, in particular to Evgueni Smirnov for his technical support and Peter Vermeer for initiating the project and enabling collaboration with the University College Maastricht.

8. REFERENCES

- [1] Bakhshinategh, B., Spanakis, G., Zaiane, O. R., & ElAtia, S. (2017). A Course Recommender System based on Graduating Attributes. In *CSEDU* (1) (pp. 347-354).
- [2] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan), 993-1022.
- [3] Bydžovská, H. (2016). Course Enrollment Recommender System. *International Educational Data Mining Society*.
- [4] Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1), 1.
- [5] Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian data analysis*. Chapman and Hall/CRC.
- [6] Griffiths, T. L., & Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1), 5228-5235.
- [7] Gulzar, Z., & Leema, A. A. (2016). An ontology based approach for exploring knowledge in networking domain. In *2016 International Conference on Inventive Computation Technologies (ICICT)* (Vol. 1, pp. 1-6). IEEE.
- [8] Gulzar, Z., Leema, A. A., & Deepak, G. (2018). PCRS: Personalized course recommender system based on hybrid approach. *Procedia Computer Science*, 125, 518-524.
- [9] Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*, Springer Series in Statistics.
- [10] Hornik, K., & Grün, B. (2011). topicmodels: An R package for fitting topic models. *Journal of Statistical Software*, 40(13), 1-30.
- [11] Jiang, W., Pardos, Z. A., & Wei, Q. (2019, March). Goal-based Course Recommendation. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (pp. 36-45). ACM.
- [12] Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22, 79-86.
- [13] Meyer, D., Hornik, K., & Feinerer, I. (2008). Text mining infrastructure in R. *Journal of statistical software*, 25(5), 1-54. URL <http://www.jstatsoft.org/v25/i05/>.
- [14] Phan, X. H., Nguyen, L. M., & Horiguchi, S. (2008). Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on World Wide Web* (pp. 91-100). ACM.
- [15] Roberts, M. E., Stewart, B. M., Tingley, D., Lucas, C., Leder-Luis, J., Gadarian, S. K. & Rand, D. G. (2014). Structural topic models for open-ended survey response. *American Journal of Political Science*, 58(4), 1064-1082.
- [16] Team, R. C. (2013). A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing. URL <http://www.R-project.org>.
- [17] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267-288.
- [18] Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, 59(10), 1-23. doi:10.18637/jss.v059.i10.
- [19] Wickham, H., Francois, R., Henry, L., & Müller, K. (2015). dplyr: A grammar of data manipulation.
- [20] Wickham, H. (2016). *ggplot2: elegant graphics for data analysis*. Springer.
- [21] Zaki, M. J. (2001). SPADE: Efficient algorithm for mining frequent sequences. *Machine learning*, 42(1-2), 31-60.

Using a Glicko-based Algorithm to Measure In-Course Learning

Rachel Reddick
Coursera, Inc.
381 E Evelyn Ave
Mountain View, California 94041
reddick@coursera.org

ABSTRACT

One significant challenge in the field of measuring ability is measuring the current ability of a learner while they are learning. Many forms of inference become computationally complex in the presence of time-dependent learner ability, and are not feasible to implement in an online context. In this paper, we demonstrate an approach which can estimate learner skill over time even in the presence of large data sets. We use a rating system derived from the Elo rating system and its relatives, which are commonly used in chess and sports tournaments. A learner's submission of a course assignment is interpreted as a single match. We apply this approach to Coursera's online learning platform, which includes millions of learners who have submitted assignments tens of millions of times in over 3000 courses. We demonstrate that this provides reliable estimates of item difficulty and learner ability. Finally, we address how this scoring framework may be used as a basis for various applications that account for a learner's ability, such as adaptive diagnostic tests and personalized recommendations.

Keywords

Adaptive learning, student modeling, knowledge tracing, student ability estimation

1. INTRODUCTION

A requirement of any adaptive learning system is the simultaneous understanding of learner skill and item difficulty. Such measurements also enable ordering content by measured difficulty and recommending content and assessments appropriate for a learner's degree of skill, among other applications.

Item Response Theory, one common method for obtaining skill estimates in a testing context, requires assuming a fixed skill for the learner. While valid during a single exam, this assumption fails for learners who are learning during a course. Some techniques can rigorously handle

skills that change over time, such as knowledge tracing [2] and performance factor analysis [6], but they are computationally intensive. For Coursera's dataset, applying these techniques would require computing results for millions of learners across thousands of courses. Further, because learners in an online platform can benefit from visibility into their own skills, online updates are desirable. Under these conditions, many approaches become computationally intractable.

To address this problem, we turned to the Elo and related rating systems, which are commonly used in chess tournaments and for team ratings in many sports [7, 9]. Following the example of [7], we treat learners and course items as players in a tournament. Each attempt by a learner at passing an item is a match.

In this paper, we demonstrate the application of the Glicko rating system [3], a variant of the Elo rating system which incorporates uncertainty, to Coursera's unique dataset. We maintain a focus on the most well-measured skills on our platform. We show that we can obtain reasonable and reliable estimates of learner skill and item difficulty. Finally, we discuss planned applications and next steps.

2. BACKGROUND

Coursera is an online learning platform, which offers courses in partnership with the universities and companies that create them. In addition to single courses, Coursera also provides "specializations," which are groupings of 3-10 courses that are usually (but not always) intended to be taken in sequence. Coursera has over 3300 different courses available, covering a broad range of subjects.

Instructors who create courses on Coursera can provide a rough estimate of the difficulty ("beginner", "intermediate", "advanced"), but this is generally not enough to establish prerequisites or help a learner know if they are ready to start a course. Since the label is at the course level, skill-related nuance is lost. For example, a course may teach both intermediate statistics and introductory programming. These labels are also insufficient for a learner to determine whether it would be more valuable to them to jump in to a course or specialization halfway through, rather than start at the beginning. Therefore, it's useful to estimate content difficulty independent of instructor labels. Personalizing this support to individual learners requires estimating their degree of skill as well.

Rachel Reddick "Using a Glicko-based Algorithm to Measure In-Course Learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 754 - 759

Many other people have previously estimated content difficulty and learner skill. The most fundamental include approaches based on Item Response Theory (IRT; e.g., [1]). However, these are not suitable in Coursera’s context because they generally assume constant learner skill, as in a testing environment. Although alternatives such as Bayesian knowledge tracing [5] have potential, the methods are difficult to scale up from a few tens of thousands of item attempts to over ten million for a single skill in Coursera’s case. Further, the authors of [5] also note the confounding effect of learners with stronger skills working on more challenging problems, which systematically underestimates the difficulty of advanced content, even in some knowledge tracing approaches.

Therefore, we follow the work of [7] and consider the Elo rating system. The Elo system is commonly used in chess tournaments and other competitions. The general principle is that every player has a score, which is updated after matches. Updates will be large if the outcome of a match is unexpected. If a novice player defeats a master, the novice’s score will have a large increase and the master will have a large decrease. On the other hand, if a master defeats a novice, the updates for both will be small or negligible. This method may be applied to learners and items in courses (such as exams), where each learner and item is interpreted as a player. A new learner who “defeats” a hard item in the Machine Learning skill will gain a large increase to that skill, but not lose much from their score if they fail. Because that ability differs from other skills (such as Management), a learner should have different scores for each possible skill. Items also need to be associated with the relevant skills.

In contrast to [7], we focus on the Glicko variant of the Elo rating system [3]. This variant uses an approximated Bayesian framework, which incorporates uncertainty in the rating. This is particularly helpful for understanding learners who have not yet completed very many items. It also supports estimation of ability across populations by enabling weighted averages based on the uncertainty. The Glicko scoring system has the drawback that it can’t be easily adapted to use more complex IRT solutions, such as those for multiple choice questions with a non-zero success probability even in the case of very low skill. However, because we focus on full assignments rather than individual questions, this will not be a serious limitation.

3. DATA

Coursera has over 38 million registered learners on its platform. We draw our data from late 2014 onward.

We also have a framework for automatically tagging skills to courses. We will focus on those subjects – business, computer science, and data science – where the skills framework is best calibrated. In this two sections, we briefly describe how we tag skills to courses, and how we process the course item data that we use.

3.1 Associating Content with Skills

Based on Wikipedia’s hierarchy of topics and our own human curation, we developed a hierarchy of more than 40,000 skills, over 13,000 of which are tagged to courses in the Coursera catalog (see [8] for additional details). The most

popular of these skills are listed in Table 1, out of a total of 26 available. These broad skills all have a set of subskills within the skill hierarchy.

Individual skills are tagged to courses based on a combination of crowd-sourcing and machine learning. Learners who complete a course are asked to tell us what skills they learned. This information is used as the target variable of a machine-learning model, which estimates the likelihood of tagging based on course content features. The actual tag rate and machine learning prediction are combined to create a single relevance score, giving results for both popular courses and unpopular courses with few crowd-sourced tags.

For our broad skills of interest, we treat a skill as tagged to a course if any of its subskills in the course’s subject area (e.g., data science) are tagged to the course.

Using this approach, for this set of skills, we obtain a total of about 1400 courses tagged with skills. Of these, about 1000 are tagged with more than one skill.

3.2 Item Attempts

We define an item attempt as a learner submission. On the Coursera platform, items include exams with multiple-choice or text answers, programming assignments that require submitting code, and peer review assignments that are graded by other learners. Most, but not all, are graded. In this analysis, we include only items that are graded. Learners are typically able to retake items up to a maximum number of attempts per day.

Learners may retake an item for many reasons, from trying again to technical issues. These repeated attempts are often uninformative. For this reason, we reduce learner attempts to those that are either the first attempt or a later attempt that does not have the same pass/fail outcome. Thus, most learner-item interactions are of the form pass (with no further attempts), fail (with no further attempts), or fail followed by pass at a later time.

This approach has the secondary benefit that if the skill scores are surfaced to learners in the future, there is no longer a motive to repeatedly submit a passed item in order to game the system to get a higher score.

We also remove all attempts at items that all learners pass on the first attempt, since these items are not informative.

We currently do not estimate scores for individual questions or sub-parts within items. Using entire items has the advantage of being able to easily include atypical items, such as programming assignments. Expanding to obtain scores for individual exam questions is left for future work.

4. IMPLEMENTATION

Our implementation of the Glicko scoring system generally followed [3], with several modifications.

Most importantly, the Glicko system assumes that players encounter each other during a tournament, during which individual scores can be assumed to be roughly constant. This is the “rating period” over which matches are accu-

Table 1: Popular Skills

Skill	Number of Attempts	Number of Courses
Statistical Programming	17,840,101	157
Machine Learning	134,52,639	60
Computer Programming	12,940,557	383
Software Engineering	9,287,216	245
Artificial Intelligence	7,900,817	127
Management	6,104,244	386

mulated, and after which an update to the scores is made. However, scores may change rapidly for learners who are learning within a course. Learners watch course lectures or review supplementary material about 40% of the time between subsequent submissions. Therefore, we are forced to use a rating period that is only one “match” long – one item attempt. In this case, the update equations from [3] reduce to, for a single match:

$$\mu' = \mu + \frac{1}{1/\sigma^2 + 1/\delta^2} g(\sigma_o^2) \{s - E(s|\mu, \mu_o, \sigma_o^2)\} \quad (1)$$

$$\sigma'^2 = \left(\frac{1}{\sigma^2} + \frac{1}{\delta^2} \right)^{-1} \quad (2)$$

Where:

$$g(\sigma^2) = \frac{1}{\sqrt{1 + 3\sigma^2/\pi^2}}$$

$$E(s|\mu, \mu_o, \sigma_o^2) = \frac{1}{1 + \exp[-g(\sigma_o)(\mu - \mu_o)]}$$

$$\delta^2 = [g(\sigma_o^2)^2 E(s|\mu, \mu_o, \sigma_o^2) \{1 - E(s|\mu, \mu_o, \sigma_o^2)\}]^{-1}$$

In these expressions, μ is the initial score, μ' is the updated score, σ is the initial uncertainty in terms of standard deviation, and σ' is the updated uncertainty. The values with a subscript ‘o’ are those for the opponent. If the “player” is a learner, then the opponent is an item (and vice versa). s is the outcome of the match. For a learner, passing the item is a ‘win’, and $s = 1$, and failing is a loss with $s = 0$. The reverse is true for items.

Our difficulty scores for items are based entirely on the scoring method described above. Instructor provided difficulty labels were not used as a feature in this framework.

Importantly, the learner’s score carries over from one course to another, so long as both courses teach the same skill. For example, when a learner finishes a course and starts another, their score for the skill at the end of that course is used as their starting point for that skill in the new course. If the courses teach different skills, then the learner’s score for the new skill taught in the second course will start at zero, unaffected by their score in previously learned skills.

Finally, to obtain high-quality scores for all learners, we find that we need to run the Glicko scorer twice. In the first run, we set both learners and items to have prior scores of 0

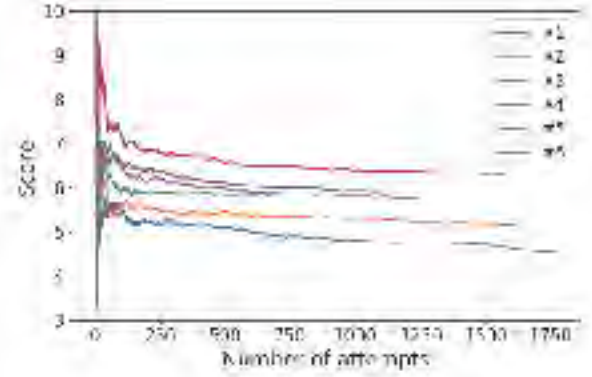


Figure 1: Scores for course items generally converge after many attempts. This example shows several items from the Practical Reinforcement Learning course, in the Machine Learning skill. The item scores rapidly converge to stable values, with a small amount of negative drift. For clarity, we only show the first few items in the course.

and uncertainties of 5, to allow large updates initially and to prevent questions with well-established scores to avoid experiencing large perturbations from new learners. This provides good estimates of item difficulty, but means that learners who took the same item early in the data, rather than late, are matched up against items whose scores may be far from reasonable values, especially for less popular content. It also means that a learner who has completed the same course, but early in our data, may not have the same score as a learner who performed in exactly the same way in the same course near the end.

Thus, all learner scores are reported based on the Glicko scorer a second time, using the item scores and uncertainties from the first run. These values for items are held fixed. The items scores for each skill are offset so that the fifth percentile item has a score of zero. Learners start with prior scores of zero and prior uncertainty of 1.0, which avoids excessive initial swings in score.

In principle, it is possible to construct a more accurate learner prior score based on information they have given us (e.g., education history) and what course they are starting. However, this could also lead to gaming of the system (by providing false information or trying a hard course first to obtain a better score), may be biased against learners with nontraditional backgrounds, and introduces additional complexity.

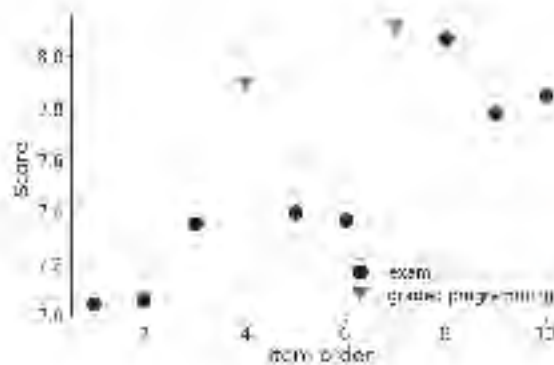


Figure 2: Comparison of final item scores within a single course (in the Machine Learning skill). As expected, later items are generally more difficult than earlier ones. The method also successfully captures differences in difficulty among different types of items – graded programming assignments tend to be more challenging than exams.

5. RESULTS

We show the convergence results for a few items in a typical course in Fig. 1. This is the Practical Reinforcement Learning course, within the Machine Learning skill. As we might expect, items later in the course are typically more challenging, but not always. This reflects variation in difficulty of sub-topics within the course.

Although the item scores clearly converge within the first few hundred attempts, there is also a long-term negative drift in the scores. This is a known occurrence within Elo frameworks (for examples in chess, see [4, 10]). In the case of chess, players generally begin playing as novices and stop playing as masters, with a higher skill than they started. The pressure of new players always starting with low scores leads to ratings deflation over time, and chess tournament rating systems often include corrective factors for this reason. In our case, a similar effect occurs because learners generally start with lower scores than those of the items.

The degree of drift over a few hundred attempts is typically of the same order as the estimated uncertainty (around 0.2 for the first item in the example course). In the future, we may add a correction for this drift effect, but it does not currently have a strong effect on our results.

For clarity, we note that convergence is only expected or desirable for item scores. Items within courses are generally not modified over time, and therefore, the difficulty of the item should not change. In contrast, learner scores can be expected to change rapidly as they progress through course content.

When we examine the final item scores more closely, we find that there are some differences by item type. Programming assignments are often more difficult than regular exams (see Fig. 2), in agreement with our expectations for these items.

Even with this variation and within-topic variability, we find that difficulty does generally increase from the beginning to

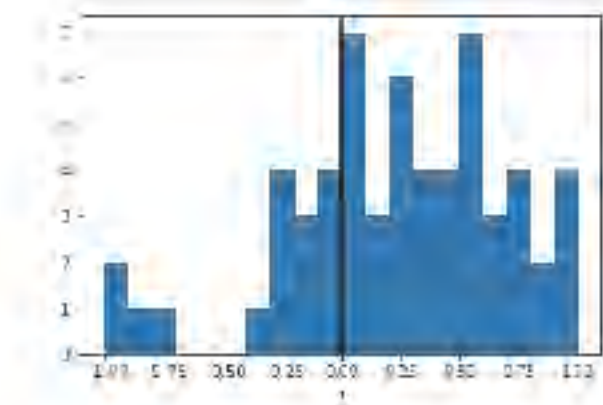


Figure 3: Histogram showing the distribution of the correlation between item difficulty and item order for each course in the Machine Learning skill. Extreme correlation (or anti-correlation) generally comes from courses with few items.

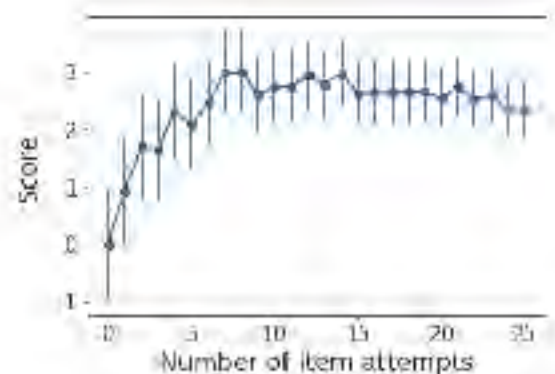


Figure 4: learner score while progressing through a course. This plot shows a learner progressing through a Machine Learning course, and completing it. Note the occasional small drop in score when the learner initially fails an item. Error bars are $\pm 1\sigma$.

end of a course. Across all skills, the mean correlation between item order and item difficulty within a course is 0.25. The distribution of correlations for a single skill, Machine Learning, is shown in Fig. 3.

We also find that scores across different courses make sense. For example, based on median item score within the Machine Learning skill, the “Google Cloud Platform Big Data and Machine Learning Fundamentals” course is the easiest. This is the first course in a sequence of courses which are intended to be introductory. Conversely, the hardest course is “Probabilistic Graphical Models 3: Learning”, the final course in a series of advanced Machine Learning courses.

A representative example of a learner gaining the Machine Learning skill is shown in Fig. 4. The learner rapidly increases in score to match the initial difficulty of the course, followed by more incremental increases later on. This is

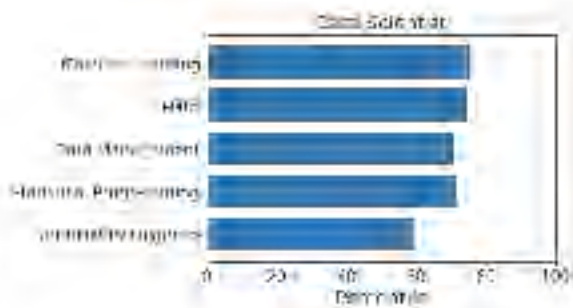


Figure 5: A profile showing the top 5 most relevant skills for a data scientist and the typical degree of proficiency. In this example, the skill score is shown as a percentile compared to the rest of the learners on Coursera.

fairly typical of learners moving through a single course.

6. FUTURE APPLICATIONS

These difficulty and ability scores, in combination with our skills tagging framework, unlocks many applications:

1. **Learner skill profiles.** These surface a summary of each learner's measured abilities. Providing this information to learners would enable them to better understand their own skills and how much they have learned so far. These profiles would update online to immediately reflect recently submitted items.
2. **Career skill profiles.** In contrast to a learner skill profile, a career skill profile shows the skills important to a career and the necessary degree of proficiency. These profiles can be built based on the skills of learners on Coursera who are already in these careers. Then these profiles can be compared against a learner profile, to allow the learner to see what they still need to enter their desired career.
3. **Recommendations by difficulty.** Given knowledge of a learner's current and desired ability in a skill, we can recommend courses that teach that skill which are at the right level for that individual.
4. **Adaptive diagnostics.** With difficulty scores for course items, it is possible to extract questions from course content and use them as the question bank for an adaptive diagnostic, which provides questions close to a test taker's estimated skill. If learners take diagnostics as a pre-test, before taking any courses, this can support the recommendations by difficulty discussed above.
5. **Review recommendations.** If a learner struggles in part of a course, a recommendation to review could find lectures or reading, potentially in another course entirely, which teaches the same skill at an easier difficulty.

We show an early example of a career skill profile for a Data Scientist in Fig. 5. These initial results align with our expectations, in that more fundamental skills for the career

(e.g., Machine Learning) are required at a higher relative skill than the more niche skill of Artificial Intelligence.

We would expect this approach to generalize well to other data sets, if the fundamental data can be constructed as attempts by learners at passing items.

For simplicity, our current approach omits the addition of uncertainty for time passed since the last scoring update. In the original Glicko model, this incorporates how a player's skills may have increased due to practice or decayed due to lack of use over time. In the future, it may be valuable to restore this factor, with additional uncertainty for learners who have spent a significant time perusing course material, or who have spent a long period away from course content.

Similarly, this approach ignores cases where multiple skills are related or relevant. For example, Machine Learning and Artificial Intelligence are closely related. However, because we treat each skill in isolation, a learner who has only taken courses in Machine Learning will not have an estimated score for Artificial Intelligence. Similarly, an item in a course may require more than one skill to pass the item. In future work, we will consider multiple-skill approaches that can address these cases.

7. CONCLUSIONS

In sum, we find that using the Glicko rating system in an educational context produces reasonable scores for learners and items. The approach successfully mitigates the potential bias from more skilled learners encountering more difficult items.

We plan to use these results in many applications in the future to improve the learning experience for learners on Coursera.

8. ACKNOWLEDGMENTS

I would like to thank my colleagues, Vinod Bakthavachalam and Alan Hickey for providing valuable input on the methods and feedback on the contents of this paper. I would also like to thank Emily Sands for her feedback and support.

9. REFERENCES

- [1] H. T. Binh and B. T. Duy. Student ability estimation based on irt. *2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science*, pages 56–61, 2016.
- [2] A. Corbett and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4:253–278, 1994.
- [3] M. E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Appl. Statist.*, 48:377–394, 1999.
- [4] H. Goldowsky. A conversation with mark glickman. *Chess Life*, Oct 2006.
- [5] J. González-Brenes, Y. Huang, and P. Brusilovsky. General features in knowledge tracing: Applications to multiple subskills, temporal item response theory, and expert knowledge. *Proceedings of the 7th International*

- Conference on Educational Data Mining*, pages 84–91, 2014.
- [6] P. Pavlik, H. Cen, and K. Koedinger. Performance factors analysis-a new alternative to knowledge tracing. In *Proc. of Artificial Intelligence in Education*. IOS Press, 2009.
 - [7] R. Pelánek. Applications of the elo rating system in adaptive educational systems. *Computers and Education*, 2016.
 - [8] E. G. Sands. How our skills graph is helping learners find the right content to reach their goals, July 2018.
 - [9] N. Silver. Our nba player projections are ready for 2018-19. *FiveThirtyEight*, June 2018.
 - [10] World Chess Federation. *FIDE Rating Regulations*.

Affect detection in home-based educational software for young children

Roger Smeets
Scula
roger@scula.com

Francette Broekman
Scula
francette@scula.com

Eric Bouwers
Scula
eric@scula.com

ABSTRACT

Research on automated affect detection in educational software using play log data has shown promising results. Yet most studies use classroom-based software designed for adolescents or adults. In this paper, we aim to detect affection in an online educational platform primarily aimed at home use by young children. This presents two challenges: we have to rely on a self-report instrument of affect that users can utilize at home, and we have to make sure that this instrument is properly understood by children. To this end, we developed and validated an emoticon-based self-report instrument to derive ground-truth labels of four emotions: Joy, frustration, confusion, and boredom. Training a number of different classifiers for automated affect detection yields promising results, in particular for detecting joy and frustration.

1. INTRODUCTION

It is by now well established that student engagement is an important correlate of learning efficacy, academic performance, and even long-term professional achievements [1, 2]. Consequently, monitoring student engagement is of key interest, so that appropriate interventions can be applied when necessary. Educational software holds a strong promise in this regard, as it enables the collection of troves of data that can be mined for engagement and learning patterns.

Previous research has demonstrated that student engagement is a multifaceted concept, typically encompassing behavioral, cognitive, and emotional aspects [5–7, 14]. There is a large body of empirical literature that aims to identify these facets of engagement during educational software use. Emotional engagement – or affect detection – in particular has garnered a lot of recent interest. Previous studies in this area have collected data from sensors and play logs to construct and identify metrics that can detect different types of emotions (and changes therein) [1, 3–5, 12].

Our objective in the current paper is to explore the possibility for affect detection based on play logs in a particular

online learning platform, called Scula. This platform embodies two traits that inhibit a straightforward application of previous research. First, Scula is primarily developed for use at home, whereas virtually all previous research considers educational software used in the classroom. Second, Scula is used by children aged 4-12, whereas previous studies mostly consider adolescents or adults.

In order to train models for automated affect detection, we first have to collect instances (i.e. ‘ground truth labels’) of students’ emotions during software use. Ideally, these data are collected during normal use of the product, i.e. when students play at home. This in turn requires a self-report instrument that can be understood and used by children. In particular, the instrument should take into account that young children cannot read, and that identifying their own emotions might be problematic.

To address these issues, we first design and validate a self-report instrument of affect that is based on emoticons. In particular, we aim to identify four emotions: Joy, boredom, confusion, and frustration [1, 3]. We then use this instrument to collect ground truth labels of affect during software use. Finally, we construct a number of features from the play logs and correlate these with the labeled emotions, building four separate affect detectors (one for each emotion).

2. THE ONLINE LEARNING PLATFORM

The software used in this study is an online educational gaming platform for K-6 children (ages 4-12) in the Netherlands, called Scula. Scula’s primary business model is aimed at home-use, i.e. paid subscriptions are sold to parents. The platform is available through web as well as via a native app (iOS and Android). It is set up primarily along 8 education classes that correspond with the grades in kindergarten (KDG), pre-school (pre-K), and elementary school. Children can play a variety of subjects, both curriculum based (e.g. math or spelling) as well as outside the curriculum (e.g. social skills or 21st century skill).

Within a subject, students can choose a topic, within which they can choose a mission, that itself is typically subdivided into multiple (hierarchical) levels.¹ One level typically consists of ten questions that have to be answered correctly be-

¹For example, grade 3 has the subject geography, containing the topic ‘weather’, which has two missions. The first is ‘seasons’, consisting of five levels, and the second is an instruction video about rain.

Roger Smeets, Francette Broekman and Eric Bouwers "Affect detection in home-based educational software for young children" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 760 - 765

fore the level is completed and the next one is unlocked. Importantly, students have complete freedom regarding what to play. The only restriction is that within a mission, levels have to be played in a predetermined order.

There are various different question formats, such as multiple choice questions, puzzles, open-answer questions, bubble poppers (popping foam bubbles containing answers), and catapult questions (shoot at answers with a catapult). If a question is answered incorrectly, the right answer is highlighted, and the question is moved to the end of the queue, so that the user has to answer it again after cycling through the remaining questions. At any point during playtime, a user can choose to quit the current level. She can always go back to that level and pick up where she left off.

When answering a question correctly, the user is rewarded with a coin. There are also numerous ways to earn bonus coins, e.g. when participating in a thematic campaign. Coins can be used to buy new avatars, or non-digital goodies from the goodie shop (e.g. cinema vouchers, toy-store vouchers, or paper-cut animals). During play, users also collect experience points, which gradually increases their experience level. Each experience level unlocks new benefits, such as new avatars in the avatar shop.

When users log onto the platform, they land on the home screen depicted in Figure 1. Here, they can navigate between the different subjects, but also to their playing statistics, to the Scula shop – where they can buy avatars and goodies with the collected coins – and to the ‘social’ screen – where they can search for friends, send and accept friend requests, and send or read messages to and from friends and parents. The top of the screen shows their experience level, their experience points, and the number of coins collected. Finally, the home screen also offers play recommendations in the three tiles in the top bar.

3. PREVIOUS RESEARCH

3.1 Log-based affect detection

Previous research on automated affect detection has used both sensor data and play log-based data to find correlates of affect. The educational software used in this study does not utilize sensors, so that we have to rely on the latter approach. Below we briefly review a number of studies that have successfully done so before.

In a study involving an automated tutoring system, D’mello et al. [5] use a combination of conversation-based log metrics and self and third-person reports – by one peer and two expert judges – of affective states. Their results show that machine predictions of affect are on par with those of ‘novice’ judges (self and peer) but inferior to those of the trained judges.

Contati and MacLaren [4] and Sabourin et al. [12] adopt a (dynamic) Bayesian network approach to develop a causal model linking user goals and (inter)actions to affective states. The resulting models significantly outperform default class prediction, and further allow the researchers to infer user play-goals.



Figure 1: Scula home screen (web)

Lee et al. [8] show that persistent confusion negatively correlates with learning achievement, yet that resolved confusion has a positive effect. Botelho et al. [3] resort to Recurrent Neural Networks to improve the performance of these models. Unlike previous studies, they combine all emotions in one detector, rather than training one detector per emotion.

3.2 Affect detection for young children

All the studies discussed above share two traits: First, they all involve educational software that is used in a classroom setting, and second, the study populations are usually either adolescents or adults. Yet in this paper we consider educational software designed for home use by children. This means that collecting data in a controlled setting – such as a classroom or lab – would interfere too much with the intended user experience. As a result, affect labeling has to happen at home, using self-reports. Yet this creates another challenge, which is that (young) children cannot be expected to provide reliable survey feedback [9].

A number of studies have developed instruments for usability testing of software products with children [10, 11]. This research stream restricts the notion of child engagement or satisfaction to “fun”. One validated instrument that has been successfully applied in this context is the *smiley-o-meter*. This is essentially a 5-point Likert scale with emoticons capturing the intensity of fun (vs lack of fun).

However, Padilla-Zea et al. [13] find that this instrument is not properly understood by young children (ages 3-5), since they are not able to grade the strength or intensity of a feeling. Yet their tests demonstrate that young children *are* able to differentiate between different *types* of emotions, based on different emoticons.

Our approach toward affect labeling mirrors that of Padilla-Zea et al.[13]: Instead of aiming to capture the *intensity* of one emotion, we want students to identify the relevant *type* out of several emotions. To this end, we develop an emoticon-based instrument, while taking into account that the youngest respondents cannot read.

Following Baker et al. [1] and Botelho et al. [3], we aim to capture four affective states: Joy, confusion, frustration,

and boredom.² However, before putting the emoticon-based instrument to use, we first have to ensure that we design a set of emoticons whose meaning is understood by the students. This design and validation process is described in the next section. Section 5 then discusses our initial efforts towards automated affect detection, using this instrument to collect ground-truth labels.

4. AFFECT INSTRUMENT VALIDATION

4.1 Method

We first designed six sets of three emoticons, or 18 in total. The emoticons in the first four sets were designed to capture joy, confusion, frustration, and boredom. We added two more sets to represent surprise and sadness. We then proceeded in three steps.

In step one (the offline adult test), we presented each of the six sets to 14 adults, asking them (1) to express the emotion they inferred from each of the 18 emoticons, and (2) to choose the emoticon (out of three) they found most fitting for each of the six emotions. Achieving consistency across feedback from adults served as a lower threshold in this case. Eventually, this step resulted in six emoticons preferred by adults, one for each affect type.

In step two (the offline child test), we presented these six emoticons to 23 children aged 4-12 in one-on-one sessions, asking them (1) to express the emotion they inferred from each of them, and (2) to choose the emoticon (out of six) they found most fitting for a particular scenario. This step was intended to flag potential interpretation differences between adults and children, allowing for a redesign if necessary. Eventually, this step resulted in a set of emoticons preferred by children.

In step three (the online child test), we used the emoticons from step two to develop a new online ‘affect mission’ with two levels on the Ssula platform. In the first level, children were presented each of the emoticons individually, while being asked to answer the following question: “Look at the character in the picture. What do you see?” They were then offered four answer options, one of which contained the intended meaning of the emoticon.³

In the second level of the mission, users were asked questions of the type “Which of the characters is [...]?” where [...] was substituted for a particular emotion. They were then shown five emoticons, one of which contained the one we intended to actually capture the emotion. For children aged

²To be precise, these studies aim to capture “engaged concentration” rather than “joy”. However, in designing the instrument we ran into difficulties capturing the former. Instead, we opted for “joy” as a substitute positive affective state that is more easily understood by children.

³An alternative setup would have been to ask children open-answer questions. However, since our youngest users are not able to write, this option is unfeasible. Furthermore, our experience with these types of questions is that many children abuse this freedom of answering by providing nonsensical or offensive answers, increasing the effort of identifying relevant answers. Nonetheless, we attempted to ensure that one or two of the alternative answer options served as detractors.

Table 1: Respondents by grade level

Grade level	# Respondents	Respondent share
KDG	1169	5.4
Pre-K	1972	9.1
Grade 1	2965	13.7
Grade 2	2421	11.1
Grade 3	3222	14.8
Grade 4	3411	15.7
Grade 5	3570	16.4
Grade 6	2985	13.7

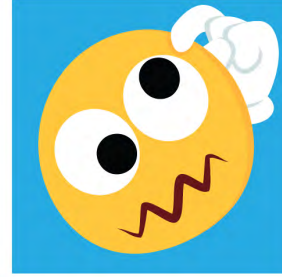


Figure 2: Initial confusion emoticon

4-7 we provided voice-recordings of both the question (in both levels), as well as the answer options (in level one).

The two mission levels were live on the platform for one week (from November 19th 2018 to November 26th 2018). During that time period, we collected responses from 21715 unique players. Table 1 shows the distribution of respondents by grade level. Although respondent shares are notably lower in the bottom two grades, they are representative of the user population across grades.

4.2 Results

The offline adult test resulted in one emoticon per affective state that was preferred by adults. In the offline child test, the emoticons for joy and boredom were correctly interpreted by children. The same holds for frustration, although most children identified this as anger instead. However, given their relatedness, as well as the fact that young children are generally not familiar yet with the notion of frustration, we deem these as interchangeable.

However, confusion was not well identified. Figure 2 depicts the original emoticon that was used. When asked what the character was feeling, typical responses were ‘nauseous’, ‘funny’, and ‘dizzy’. Further probing revealed that both the wiggly mouth as well as the cross eyes confused children’s interpretation regarding the intended meaning.

We used this feedback to redesign the confusion emoticon. In particular, two new designs were made. Additionally, we also designed an additional emoticon for frustration, in order to verify that the response ‘angry’ was not driven by the emoticon but rather by children’s lack of awareness of the concept of frustration.

All in all, we ended up with a set of eight emoticons, pre-

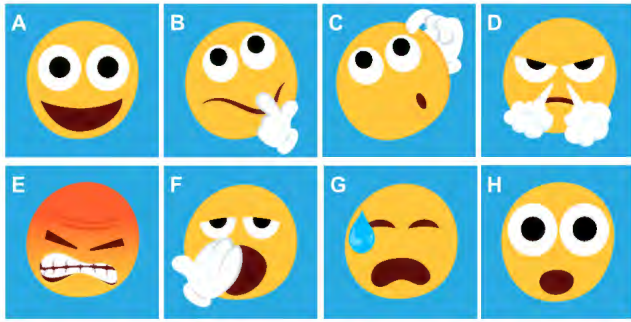


Figure 3: Final test emoticons

sented in Figure 3: Joy (A), thoughtfulness (B), confusion (C), anger (D), frustration (E), boredom (F), sadness (G), and surprise (H). Recall that we are primarily interested in capturing joy, confusion, frustration, and boredom. We had two emoticons for confusion (B and C) and frustration (D and E). We used the results of the online child tests to choose the preferred/optimal one. The emoticons for surprise and sadness were used as additional detractors in the online child test.

Using the data from the online child test, we compare the share of respondents that chose a particular answer option. Chi-squared tests show that the distribution of response shares on all questions in all grade levels are significantly different from uniform, indicating that *at least* one answer received a disproportionate share of responses. Computing Chi-squared statistics on all answer pairs per question reveals there is only one instance where *two* answer options are equally preferred: This concerns the level one question regarding the confused emoticon (C) in kindergarten.

The four answer options presented to users in this case are ‘struggling’, ‘confused’, ‘surprised’, and ‘sad’, with ‘struggling’ added as a detractor. Indeed, in Kindergarten the detractor received a sufficient number of responses so as to not be statistically differentiable from the ‘confused’ answer option. Yet for present purposes, the distinction between struggling and confused is negligible. Both these emotions correspond to what we aim to capture, which is that the user is having difficulties understanding or answering questions.

In all of the other cases, the preferred answer option corresponds to the intended answer option. The degree of variation across answer options is notably higher in the lower grades. Figure 4 illustrates this. It shows the response shares across the five emoticon options for the question ‘Which character is bored?’ Although the bored emoticon (F) receives the highest response shares in all grades, the variation is notably greater in the bottom two grades, with the angry emoticon (D) as a close second.

Finally, the online child test also enables us to identify the preferred emoticon for confusion and frustration. In all grades, emoticon C is preferred for confusion, and emoticon E is preferred for frustration. An added benefit is that the primary detractor for Kindergarten and Pre-K, i.e. emoticon D, is not part of the final set.

Table 2: Answer response shares per trigger

Answer	Level completed	Level interrupted	Navigating
happy	54.4	28.5*	33.2*
bored	15.4	14.8	11.9*
confused	9.6	9.7	6.7*
frustrated	9.5	18*	8.8
not-answered	11.1	29.1*	39.5*

* Response share different from quiz-completed ($p < 0.05$)

5. AFFECT DETECTION

5.1 Method

Using emoticons A, C, E, and F in Figure 3, we designed a modal to prompt users for feedback regarding their affective state during playtime. Since we are targeting young children whose parents pay for the product, we did not want to excessively disrupt their play experience. We therefore present the modal following three specific triggers (with a maximum of three modals per week):

1. When a user finishes a mission-level in one attempt
2. When a user prematurely interrupts a mission-level for the first time
3. When a user has been navigating the platform without playing a mission-level for three consecutive minutes

In all three cases, users are shown a modal that prompts them for their affective state. Their response is logged, together with the relevant trigger, a timestamp, an id of the particular gamesession⁴ that they are playing (in case of the first two triggers), or the url that they were visiting when they were shown the modal (in case of the third trigger). Users have the option to close the modal without providing an answer, which is logged as well.

In Section 5.2, we first compare the affect response distribution following the first trigger with that following the other two. The first trigger serves as the benchmark case: We expect that users are predominantly happy when they finish a mission-level, and that the two other triggers typically correlate more strongly with a lack of enjoyment.

In Section 5.3, we then present the results for four affect detectors (one for each affect state) for users playing the subject *math*. We construct 51 mission-level features (such as the share of correct questions, the total number of questions answered, or the minimum, median, maximum and standard deviation of answer time per question) to serve as inputs, with the collected affect responses serving as outputs. We trained the following models (hyper parameters within parentheses):

- Logistic regression with elastic-net regularization (mixing parameter between L1 and L2 regularization, and regularization weight parameter) [glmnet]

⁴A gamesession is a particular play instance of a mission-level by a student.

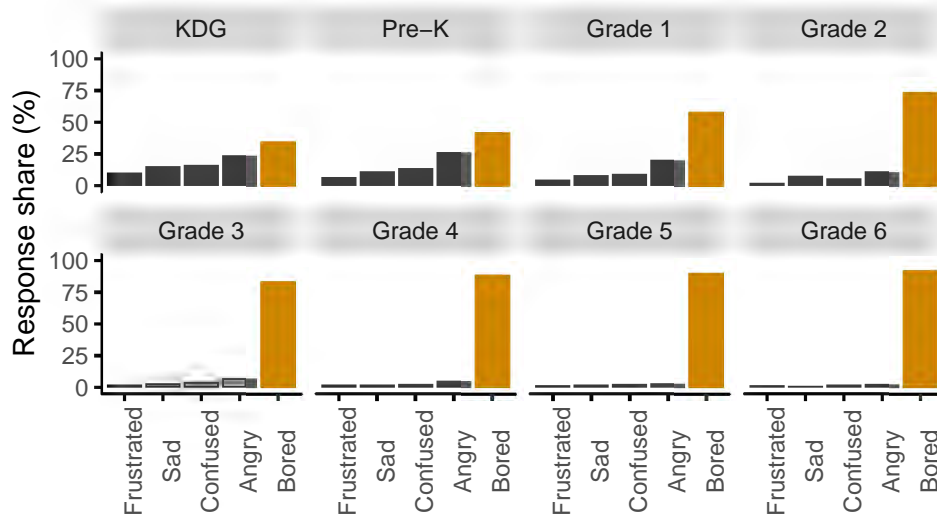


Figure 4: Which character is bored? (intended answer in orange)

- C5.0 classification tree with boosting (tree vs rule-based model, number of boosting iterations, and without or without feature selection) [**C5.0**]
- Naive Bayes classifier (Laplace correction and bandwidth adjustment) [**nb**]
- Random forest (the number of selected features to try during bagging) [**rf**]
- Gradient boosting machine (number of boosting iterations and maximum tree depth) [**gbm**]

Table 3: Model training results

Affect state	Model	Kappa	AUC
happy	gbm	0.212	0.643
bored	C5.0	0.051	0.541
confused	rf	0.077	0.617
frustrated	glmnet	0.173	0.669

Following Baker et al. we use 5-fold student level cross-validation to guide hyper-parameter tuning of the model, and conduct down-sampling to achieve response balance during cross-validation [1]. In terms of hyperparameters, we perform grid searches to determine the optimal parameters during cross-validation, and use the optimal model to compute performance on the (held out) test set. The objective metric for model training is Kappa.

5.2 Trigger comparisons

Table 2 presents the share of responses per answer-type and per trigger. In all three cases, the dominant answer option is ‘happy’. Yet the share is notably higher when users complete a level than when they interrupt it or when they are navigating. Conversely, the share of frustrated users is notably higher when a level is interrupted. There is little difference in boredom and confusion between completing and interrupting a level, however. Finally, players are much more likely to ignore the modal when they are navigating or after they interrupt a level, compared to when they complete a level.

It is noteworthy that a non-negligible share of users chooses to ignore the modal, in particular when interrupting a level or after navigating for a while. Given the difference with the ‘level completed’ share on this response, this *could* be indicative of a lack of enjoyment as well, yet without more detailed user feedback this is impossible to verify.

These patterns are by and large replicated across grades and subjects (for the level completed and interrupted triggers).

The most notable differences are that players in kindergarten and pre-school (1) more often choose to ignore the modal (regardless of the trigger), and (2) are less frustrated when interrupting a mission-level.

5.3 Affect detectors

Table 3 presents the model results in terms of Kappa and AUC.⁵ The detector for boredom is weakest, followed by that for frustration, confusion, and boredom, respectively. This partly reinforces the results presented in Table 2, where we observed that happiness and frustration exhibit most variation between the level completed and interrupted triggers.

Table 4 shows the top-3 features per detector. The share of correctly answered questions appears in three of the four detectors. This is also true for the response time between the moment the modal is shown, and when it is answered. Finally, note that *happy* and *frustrated* share two of the top-three features. This could indicate that these two emotions are two sides of the same coin.

6. DISCUSSION & CONCLUSION

In this paper we have developed and validated an instrument for affect detection that can be used during home-based educational software use by (young) children. Consistent with previous research, we find that children aged 4-12 are able to

⁵The model performance statistics reported in Table 3 are based on a test set with the original, non-resampled response distributions. The test data are again completely disjoint from the training data at the student

Table 4: Top-3 model features

happy	share of correctly answered questions response time to modal trigger that generates the modal
bored	player grade level share of correctly answered questions a repetitive streak of calculator questions
confused	response time to modal the type of questions that was answered last the average difficulty of questions in the level
frustrated	the difference between user performance and average question difficulty share of correctly answered questions response time to modal

differentiate between four different affective states (joy, boredom, confusion, and frustration) based on a set of emoticons.

We find that users in our study that complete a mission-level are predominantly happy, whereas those that interrupt it are predominantly unhappy. Furthermore, this unhappiness mostly reveals itself as frustration. For users that have been navigating for an extended period of time, the conclusion are less clear, in particular given the high share of users which ignore the modal.

In comparison to existing studies on affect detection in educational software, we find that the response distribution across affect states – reported in Table 2 – is much less skewed. For example, Baker et al. (2012) find that close to 85% of their responses capture engaged concentration, and just 0.9% confusion [1]. The differences with the results reported in this study could be driven by a variety of factors, most notably the fact that our target population includes children instead of adults/adolescents, and the fact that students are playing at home rather than in the classroom. More generally, recall that students are completely free to pick and choose what they play on the platform. This is very different from many of the previous studies, where software is used in the classroom and as part of the curriculum, including instruction. At the very least, it illustrates that the findings in this literature cannot simply be extended to other contexts.

Compared to Baker et al. [1], the performance of our automated affect detectors is quite a bit weaker. In part, these differences might be driven by the different number of model features (51 vs 258) and in target populations, i.e. children vs adolescents. Yet another explanation might be that our emoticons leave more room for interpretation differences than text surveys. As a result, out affect identification might be less precise than in earlier studies.

Going forward, the ultimate goal is to be able to detect affect during playtime, in order to provide meaningful and effective interventions. The processes and results reported in this paper are a first step towards that goal. The best performing models are still weaker than those found in previous research. However, the models for detecting joy and frustration perform notably better than chance. This suggests that it is worthwhile to put more effort into feature construction and

algorithm selection for these detectors.

7. REFERENCES

- [1] Baker, R.S. et al. 2012. Towards sensor-free affect detection in cognitive tutor algebra. *Educational data mining 2012, june 19-21, proceedings* (2012).
- [2] Baker, R.S.J.d. et al. 2010. Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive-affective states during interactions with three different computer-based learning environments. *Int. J. Hum.-Comput. Stud.* 68, 4 (Apr. 2010), 223–241.
- [3] Botelho, A.F. et al. 2017. Improving sensor-free affect detection using deep learning. *Artificial intelligence in education AIED 2017, june 28 - july 1* (2017), 40–51.
- [4] Conati, C. and Maclaren, H. 2009. Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction*. (2009), 267–303.
- [5] D'Mello, S.K. et al. 2008. Automatic detection of learner's affect from conversational cues. *User Modeling and User-Adapted Interaction*. 18, 1-2 (Feb. 2008), 45–80.
- [6] Fredricks, J.A. et al. School engagement: Potential of the concept, state of the evidence. *Review of Educational Research*. 74, 1, 59–109.
- [7] Hershkovitz, A. and Nachmias, R. 2008. Developing a log-based motivation measuring tool. *Educational data mining 2008, june 20-21, proceedings* (2008), 226–233.
- [8] Lee, D.M.C. et al. 2011. Exploring the relationship between novice programmer confusion and achievement. *4th international conference on affective computing and intelligent interaction* (Memphis, TN, 2011), 175–184.
- [9] Markopoulos, P. et al. 2008. *Evaluating children's interactive products: Principles and practices for interaction designers*. Morgan Kaufmann Publishers Inc.
- [10] Read, J. and Macfarlane, S. 2002. Endurability, engagement and expectations: Measuring children's fun. *Interaction design and children, shaker publishing* (2002), 1–23.
- [11] Read, J.C. 2008. Validating the fun toolkit: An instrument for measuring children's opinions of technology. *Cogn. Technol. Work.* 10, 2 (Mar. 2008), 119–128.
- [12] Sabourin, J. et al. 2011. Modeling learner affect with theoretically grounded dynamic bayesian networks. *Proceedings of the 4th international conference on affective computing and intelligent interaction* (2011), 286–295.
- [13] Zea, N.P. et al. 2013. A method to evaluate emotions in educational video games for children. *J. UCS.* 19, 8 (2013), 1066–1085.
- [14] Zhu, E. 2006. Interaction and cognitive engagement: An analysis of four asynchronous online discussions. *Instructional Science*. 34, 6 (Nov. 2006), 451.

Reinforcement Learning for Educational Data Mining Workshop (RL-EDM 2019)

Hamoon Azizsoltani
North Carolina State
University
Raleigh, NC, 27695
hazizso@ncsu.edu

Tiffany Barnes
North Carolina State
University
Raleigh, NC, 27695
tmbarnes@ncsu.edu

Min Chi
North Carolina State
University
Raleigh, NC, 27695
mchi@ncsu.edu

Markel Sanz Ausin
North Carolina State
University
Raleigh, NC, 27695
msanzau@ncsu.edu

Joseph Jay Williams
University of Toronto
Toronto, Ontario
williams@cs.toronto.edu

Yeo-Jin Kim
North Carolina State
University
Raleigh, NC, 27695
ykim32@ncsu.edu

ABSTRACT

Advances in the tutoring systems present a unique opportunity for effective and cost-efficient tutoring for the students. Such tutoring systems produce a large amount of educational data. Therefore, there is a growing need for data-driven learning techniques applicable to the development of Intelligent Tutoring Systems (ITSs)[4, 7, 18]. One of the goals of ITSs is to improve the traditional education methods by providing individualized tutoring for every student, from underrepresented ones to those with high competency level.

Reinforcement Learning (RL)[20] provides a data-driven approach to personalize the learning environment for each student in a cost and time effective way [11, 13]. Such an inexpensive, personalized and data-driven approach provides an opportunity for every student based on their level of competency [23]. However, several major challenges need to be addressed for the induction of an effective RL policy. For example, one of the major challenges lies in the definition of the states, actions, and rewards in the educational domain. It is known to the community that the data collected by the tutoring systems is usually noisy, and the rewards, such as learning gain, are only available with a significant amount of delay. When the immediate reward is not observable, inducing an effectiveness offline RL policy is challenging. One example of this is when we measure the performance of students in the exam they take at the end of the semester. On top of that, there are several sources of uncertainty in educational training data. For example, the student's learning state is not completely observable. At the same time, the student's interaction with the tutoring system is not deterministic which makes the training data noisy. In this

workshop, we discuss the issues to induce effective policies for educational purposes in the presence of uncertainty and delayed rewards.

1. INTRODUCTION

Since successful cases of data-driven approaches in ITS have increased substantially in the past years[2, 17], there is a growing need for researchers to share the new methods and present their work.

This workshop will provide the opportunity for researchers to 1) present their negative as well as positive results in order to learn from past experiments, 2) present and discuss the state-of-the-art techniques in RL, 3) expand the community of researchers interested in the field. In this workshop, we present full and short papers, as well as posters concerned with the application of RL in education. The research includes diverse issues in RL for ITSs such as representing learning states, defining reward functions, discovering effective actions, and introducing new ITS environments. Specifically, we cover a variety of RL topics including but not limited to:

- Tabular Reinforcement Learning[12, 3]
 - Markov Decision Processes (MDP)
 - Partially Observable MDP (POMDP)
 - Monte Carlo Simulation (MCS)
- Reinforcement Learning with approximation[19]
 - Model-based and model-free methods
 - Off-policy and on-policy methods
 - Q-learning
 - Policy Gradient algorithms
 - Actor-Critic algorithms
- Offline Policy Evaluation (Importance Sampling and variants)[21]

Hamoon Azizsoltani, Min Chi, Joseph Jay Williams, Tiffany Barnes, Markel Sanz Ausin, Yeo Jin Kim and Anna Rafferty "Reinforcement Learning for Educational Data Mining" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 766 - 767

2. RL FOR EDUCATIONAL DATA MINING 2019

RL for Educational Data Mining workshop will be held within the Educational Data Mining 2019 conference in Montreal, Canada. This workshop will be organized as a half-day mini-conference. We will start the workshop with a presentation about two ITSs developed at the Department of Computer Science at North Carolina State University. Then, we will present our recent work about the challenges of applying Deep Reinforcement Learning (DQN) [10] for Pedagogical Policy Induction. These challenges include 1) reward reshaping [1] and inferring immediate rewards from delayed rewards to induce effective pedagogical policies, 2) offline evaluation of the induced pedagogical policies [6, 9] and 3) discussion about different deep reinforcement learning algorithms and architectures such as Double DQN [22] and Double-DQN with prioritized experience replay [14], Actor-Critic methods [15, 16], Deep Deterministic Policy Gradient (DDPG) [8] and Soft Actor-Critic (SAC) [5]. Next, we will dedicate our time for the full paper presentations and discussions after each presentation. We will continue with networking and planning for future works and collaborations.

During this mini-conference, we will transfer the required knowledge to understand the best practices for conducting research with RL for Educational Data Mining. Moreover, we will discuss the challenges, limits, and future directions of RL for Educational Data Mining.

3. ACKNOWLEDGMENTS

This workshop was supported by the NSF Grants: #1522107, #1651909, #1660878 and #1726550.

4. ADDITIONAL AUTHORS

Additional authors: Anna Rafferty (Carleton College, Northfield, MN, 55057, email: arafferty@carleton.edu)

5. REFERENCES

- [1] A. G. Barto. Intrinsic motivation and reinforcement learning. In *Intrinsically motivated learning in natural and artificial systems*, pages 17–47. Springer, 2013.
- [2] M. Behrooz and B. Tiffany. Evolution of an intelligent deductive logic tutor using data-driven elements. *International Journal of Artificial Intelligence in Education*, 27(1):5–36, 2017.
- [3] M. Chi, K. VanLehn, D. Litman, and P. Jordan. An evaluation of pedagogical tutorial tactics for a natural language tutoring system: A reinforcement learning approach. *International Journal of Artificial Intelligence in Education*, 21(1-2):83–113, 2011.
- [4] K. Georgila, M. G. Core, et al. Using reinforcement learning to optimize the policies of an intelligent tutoring system for interpersonal skills training. In *AAMAS '19*, pages 737–745, 2019.
- [5] T. Haarnoja, A. Zhou, et al. Soft actor-critic algorithms and applications. *arXiv:1812.05905*, 2018.
- [6] N. Jiang and L. Li. Doubly robust off-policy value evaluation for reinforcement learning. *arXiv preprint arXiv:1511.03722*, 2015.
- [7] K. R. Koedinger, J. R. Anderson, W. H. Hadley, and M. A. Mark. Intelligent tutoring goes to school in the big city. *IJAIED*, 8:30–43, 1997.
- [8] T. P. Lillicrap, J. J. Hunt, et al. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- [9] T. Mandel, Y.-E. Liu, E. Brunskill, and Z. Popović. Offline evaluation of online reinforcement learning algorithms. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
- [10] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.
- [11] B. Mostafavi, G. Zhou, C. Lynch, M. Chi, and T. Barnes. Data-driven worked examples improve retention and completion in a logic tutor. In *Artificial Intelligence in Education*, pages 726–729. Springer, 2015.
- [12] A. Rosenfeld, M. Cohen, M. E. Taylor, and S. Kraus. Leveraging human knowledge in tabular reinforcement learning: a study of human subjects. *The Knowledge Engineering Review*, 33, 2018.
- [13] J. P. Rowe and J. C. Lester. Improving student problem solving in narrative-centered learning environments: A modular reinforcement learning framework. In *AIED*, pages 419–428. Springer, 2015.
- [14] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.
- [15] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In *ICML*, volume 37, pages 1889–1897, 2015.
- [16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [17] S. Shen, M. S. Ausin, B. Mostafavi, and M. Chi. Improving learning & reducing time: A constrained action-based reinforcement learning approach. In *UMAP*, pages 43–51. ACM, 2018.
- [18] S. Shen, B. Mostafavi, T. Barnes, and M. Chi. Exploring induced pedagogical strategies through a markov decision process framework: Lessons learned. *JEDM| Journal of Educational Data Mining*, 10(3):27–68, 2018.
- [19] S. Shen, B. Mostafavi, C. Lynch, T. Barnes, and M. Chi. Empirically evaluating the effectiveness of pomdp vs. mdp towards the pedagogical strategies induction. In *International Conference on Artificial Intelligence in Education*, pages 327–331. Springer, 2018.
- [20] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, MA, USA, 8 2018.
- [21] P. Thomas and E. Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148, 2016.
- [22] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, page 5. Phoenix, AZ, 2016.
- [23] G. Zhou, J. Wang, C. Lynch, and M. Chi. Towards closing the loop: Bridging machine-induced pedagogical policies to learning theories. In *EDM*, 2017.

Second LABBEC workshop: Learning analytics, Building bridges between the Education and the Computing communities

Sébastien Béland
Université de Montréal
sebastien.beland
@umontreal.ca

Iris Bourgault Bouthillier
Université de Montréal
iris.bourgault.bouthillier
@umontreal.ca

Michel C. Desmarais
Polytechnique Montréal
michel.desmarais@polymtl.ca

Guillaume Loignon
Université de Montréal
guillaume.loignon
@umontreal.ca

Nathalie Loye
Université de Montréal
nathalie.loye@umontreal.ca

ABSTRACT

This workshop is the second edition of LABBEC 2018 [2]. It aims to foster the exchange of ideas between the field of education measurements and education practitioners, and the EDM community. Research contributions were invited on topics related to the mutual influence between EDM and the education community. Demos of real world applications issued from EDM were presented and a discussion in the form of a round-table closed the workshop.

Keywords

Psychometrics, Educational measurements, Education settings

1. INTRODUCTION

Spawning from computer science, the Educational Data Mining (EDM) and Learning Analytics (LA) fields have generated a wealth of research over the last decade, including two yearly conferences and two scientific journals. EDM and LA typically aim to discover relations of interest by analyzing institutional or digital learning environment databases. Often so, practitioners create models borrowed from generic techniques in machine learning [1]. This preference for a domain agnostic “data science” could be construed as a lack of concern for existing educational measurement theory in the newer fields of EDM and LA. Conversely, educational measurement might be failing to engage with newer research in computing-based practices.

The first edition of the workshop was held at ITS 2018 [3]. In addition to a round-table and two demonstrations of existing Learning Analytics and EDM applications currently deployed, the half-day ses-

sion included the presentation of three 8–10 pages research papers. The program is available at this link : <http://griemetic.ca/fr/evenements/learning-analytics-building-bridges-between-the-education-and-the-computing-communities/>.

2. TOPICS OF INTEREST

The call for papers mentioned the following list of topics that were of interest, though not exclusively:

- Mutual perception of EDM/LA and educational measurement communities
- Expectations of the educational community towards EDM
- Role of psychometrics in EDM and vice-versa
- EDM tools and methods in education
- Statistical learning methods in psychometrics

3. SCHEDULE

The workshop was half-day and included four papers that were peer reviewed by two experts and the Program committee members. Two demonstrations of practical applications of EDM were presented by our invited speaker Sameer Bhatnagar from Dawson College, Polytechnique Montreal, and the Saltise Association (<https://www.saltise.ca/>).

The four research papers presented were selected from a total of six submissions. They are:

1. Björn Rudzewitz, Ramon Ziai, Florian Nuxoll, Kordula De Kuthy and Detmar Meurers. *Enhancing a Web-based Language Tutoring System with Learning Analytics*.
2. James Shing Chun Yip and Raymond Chi-Wing Wong. *Analyzing Student Performance with Personalized Study Path and Learning Trouble Ratio*.

Sébastien Béland, Iris Bourgault Bouthillier, Michel Desmarais, Guillaume Loignon and Nathalie Loye "Second LABBEC workshop: Learning analytics, Building bridges between the Education and the Computing communities}" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 768 - 769

3. Stevens Dormezil, Taghi Khoshgoftaar and Federica Robinson-Bryant. *Differentiating between Educational Data Mining and Learning Analytics: A Bibliometric Approach*.
4. Renzhe Yu, Zachary Pardos and John Scott. *Student Behavioral Embeddings and Their Relationship to Outcomes in a Collaborative Online Course*.

A round table closed the workshop.

4. REFERENCES

- [1] B. Bakhshinategh, O. R. Zaiane, S. ElAtia, and D. Ipperciel. Educational data mining applications and tasks: A survey of the last 10 years. *Education and Information Technologies*, 23(1):537–553, 2018.
- [2] S. Béland, M. C. Desmarais, and N. Loye, editors. *Learning analytics workshop: Building bridges between the Education and the Computing communities*, June 11–15, Montreal, Canada. 2018.
- [3] N. Guin and A. Kumar, editors. *ITS 2018 Workshop Proceedings*, June 11–15, Montreal, Canada. 2018.

Tutorial Proposal: Designing and Developing Open, Pedagogically-Based Predictive Models using the Moodle Analytics API

Elizabeth Dalton
Moodle Pty Ltd
2/18 Richardson St
West Perth 6005 Australia
+61 8 9467 4167
elizabeth@moodle.com

David Monllaó Olivé
Moodle Pty Ltd
2/18 Richardson St
West Perth 6005 Australia
+61 8 9467 4167
davidm@moodle.com

ABSTRACT

In this tutorial, participants explore design and development considerations for open, pedagogically-based predictive models using the Moodle Analytics API. The tutorial covers aligning models and systems with pedagogical and curriculum needs of the educational program, including interactive exercises in identifying curricular needs to inform educational data mining. We then introduce the Moodle Analytics API, including interfaces for developing Targets, Indicators, Analysers, and Time-Splitting Methods, and including hands-on exercises in which participants use the Moodle Analytics API. The tutorial concludes with a complete basic model developed in PHP using the API.

A laptop capable of installing and running a local copy of Moodle is required for full participation in this tutorial.

Keywords

Educational Data Mining, Moodle, curriculum, pedagogy, open systems

1. INTRODUCTION

In this tutorial, we first explore the educational assumptions that underlie all learning analytics models. These assumptions are often implicit and sometimes even in conflict with one another. Having identified specific goals for a program, we document the requirements for a predictive model to support those goals. As we proceed through the tutorial, we work from the requirements defined in the first exercises to implement models in PHP using the Moodle Analytics API.

This tutorial is followed up with online modules allowing participants to explore each topic in more depth, including peer-reviewed tasks with badges.

2. LITERATURE REVIEW

Over the past few decades, educational data mining systems and models have brought statistical analysis, data mining, and machine learning tools to bear on the teaching and learning process [1,2]. However, even as such predictive models become

more widespread, often they remain focused on “at risk” models that make assumptions about the definition of “student success,” such as completion of a course with a “passing” grade [3,4]. Yet there are many definitions of “student success,” and some programs are not evaluated in terms of the proportion of “passing” students, but in terms of other programmatic outcomes such as accurate identification of those with scholarly potential, influence on pro-social behaviors, or support of student-selected goals and lifelong learning skills [5]. These definitions can inform contemporary educational data mining efforts to identify new relationships and features.

The Moodle Analytics API supports a broad range of such programmatic outcomes, and provides integrated detection, modeling, and notification tools within Moodle [6]. Moodle is an open source learning management system (LMS) used by over 100 million users on over 100 thousand sites in 227 countries [7], providing a robust events and logging system, and as such is ideal as a modern platform for educational data mining, including creating, verifying and implementing predictive models.

3. AGENDA

This tutorial is designed to be delivered in a full day, and contains the following sections:

3.1 Introduction to Moodle Analytics

In this introduction, basic concepts of data analysis in the Moodle environment are briefly covered, and the features of the Moodle Analytics API are introduced.

3.2 Defining and Measuring “Quality Education”

We cannot measure what we have not defined, much less make predictions and advisements to support the quality of the learning process [8]. In this portion of the tutorial, we explore four paradigms [5] of “quality education” in terms of curriculum theory, pedagogy, stakeholders, outcomes, and potential predictors. The tradeoff between accuracy and interpretability of predictions is also discussed.

3.3 Exercise: Defining Quality Education

In this hands-on exercise, participants answer a series of questions about the nature and purpose of education and plot their responses on shared charts. The purpose of the exercise is not to determine a single definition of “quality education,” but to allow participants to unpack assumptions and make explicit the definitions they wish to support in educational data mining.

Elizabeth Dalton and David Monllaó Olivé "Designing and Developing Open, Pedagogically-Based Predictive Models using the Moodle Analytics API" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 770 - 772

3.4 Predictive Model Design Overview

Participants walk through the steps to design a predictive model, including basic data analysis and stakeholder involvement. Benefits and risks of real-time notification systems are discussed.

3.5 Introduction to the Moodle Analytics API

This overview explores the API, including functional components, specific classes to be extended, and required elements in each class.

3.6 Exercise: Prepare Git Repository

Moodle Analytics models are developed as Moodle “plugins.” Participants have an opportunity to set up a Moodle development environment and clone a skeleton Git repository for work on a model plugin.

3.7 Model Review: The Community of Inquiry Model

We review the base model shipped with Moodle Core, the “Students at risk of dropping out” model [9] based on the Community of Inquiry theoretical framework [10]. We discuss design decisions made in implementing this model and review challenges in implementing and evaluating the model for widespread use.

This presentation concludes with an opportunity to connect to a live system with sample data to observe the model in operation. Processes for evaluating learning analytics models are discussed.

3.8 Exercise: Design an Open, Ethical, Pedagogically-Based Predictive Model

Participants design a predictive analytics model “on paper,” defining the learning question or challenge to be resolved, stakeholders, user stories, the target that will be implemented, possible indicators, and a channel for discussion. Working teams present their results to the tutorial participants.

3.9 Development of Targets

The Moodle Analytics API currently supports supervised learning, requiring a training set of data with defined Targets. The Target class is examined in more detail, including the specification of outcome classes, the analyser class, constraints for valid analysables and samples, and the target calculation itself.

3.10 Insights, Notifications and Actions

Insights, Notifications and Actions are coded as part of the Target class. Insights define the text to be used in each outcome class for the model. An array of links to Actions is constructed for use by the recipient of the Notification.

3.11 Exercise: Develop a Target

Participants build on the repository created in the previous exercise to define a simple binary target.

3.12 Development of Indicators

Indicators consist of reusable classes that calculate predictors for use in models. A set of indicators are provided with Moodle Core. Indicators can be used in any model using an appropriate

Analyser (see below). Most of the current indicators calculate values by aggregating log actions and other data over a time splitting interval.

3.13 Exercise: Develop an Indicator

Participants build on the repository created in the previous exercise to define a simple indicator.

3.14 Discussion: Novel Indicators

Indicators are one of the most fertile areas for advanced educational data mining, as they can be extended to take advantage of innovations in feature discovery. Participants discuss potential novel indicators in groups and then present to the rest of the participants.

3.15 Development of Time-Splitting Methods

The time-splitting method classes determine how often to make predictions and how much data (based on time intervals) to include in each prediction.

3.16 Development of Analysers

Analyser classes define the context of the predictions, i.e. one prediction should be generated for every student in each course, or every course in each category, or every learning plan for each student. Analyser classes are usually the most complex components in the Moodle Analytics API.

3.17 Backend Engine and Algorithm Interfaces

The Moodle Analytics API currently supports two backend machine learning engines, one using php-ml and one using Python based on Tensorflow [11]. The php-ml backend provides a regression algorithm, and the python backend provides a feed-forward neural network with one single hidden layer. It is possible to extend these interfaces to make use of other backends and algorithms. Like Indicators, this is another fertile area for Educational Data Mining.

3.18 Exercise: Implement a Basic Model

In this exercise, participants work individually or in groups to extend the provided skeleton to create a simple predictive model using the API. We provide a system with realistic data on which to install and test models, and review model evaluation. The process to submit a completed model to the Moodle Plugins Database is also reviewed.

4. ONLINE MODULES

Participants may continue to explore more advanced API concepts using a series of free online modules.

5. REFERENCES

- [1] Iqbal AlShammari, Mohammed Aldhafiri, and Zaid Al-Shammari. 2013. A meta-analysis of educational data mining on improvements in learning outcomes. *College Student Journal* 47, 2 (2013), 326–333.
- [2] Alejandro Peña-Ayala. 2014. Educational data mining: A survey and a data mining-based analysis of recent works.

- Expert Systems with Applications* 41, 4 (March 2014), 1432–1462.
- [3] Bernie Dodge, John Whitmer, and James P. Frazee. 2015. Improving undergraduate student achievement in large blended courses through data-driven interventions. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, 412–413. DOI:<https://doi.org/10.1145/2723576.2723657>
 - [4] Kimberly E. Arnold and Matthew D. Pistilli. 2012. Course signals at Purdue: Using learning analytics to increase student success. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, 267–270. Retrieved June 3, 2017 from <http://dl.acm.org/citation.cfm?id=2330666>
 - [5] Michael Schiro. 2008. *Curriculum theory: Conflicting visions and enduring concerns*. Sage, Thousand Oaks.
 - [6] David Monllaó Olivé, Du Q Huynh, Mark Reynolds, Martin Dougiamas, and Damyon Wiese. 2018. A Supervised Learning framework for Learning Management Systems.
 - [7] Moodle.org: Moodle Statistics. Retrieved January 14, 2019 from <https://moodle.net/stats/>
 - [8] Dragan Gašević, Shane Dawson, and George Siemens. 2015. Let’s not forget: Learning analytics are about learning. *TechTrends* 59, 1 (2015), 64–71.
 - [9] Students at risk of dropping out - MoodleDocs. Retrieved January 14, 2019 from https://docs.moodle.org/36/en/Students_at_risk_of_droppin_g_out
 - [10] D. Randy Garrison. 2016. *E-Learning in the 21st Century: A Community of Inquiry Framework for Research and Practice* (3rd ed.). Routledge, New York.
 - [11] Machine learning backends - MoodleDocs. Retrieved January 14, 2019 from https://docs.moodle.org/dev/Machine_learning_backends

Sharing and Reusing Data and Analytic Methods with LearnSphere

Kenneth Koedinger
Carnegie Mellon University
koedinger@cmu.edu

John Stamper
Carnegie Mellon University
stamper@cs.cmu.edu

Paulo Carvalho
Carnegie Mellon University
pcarvalh@cs.cmu.edu

Philip Pavlik Jr.
University of Memphis
ppavlik@memphis.edu

Luke Eglington
University of Memphis
lggln@memphis.edu

ABSTRACT

This tutorial will explore LearnSphere, an NSF-funded, community-based repository that facilitates sharing of educational data and analytic methods. The tutorial organizers will discuss the unique research benefits that LearnSphere affords. We will focus on Tigris, a workflow tool within LearnSphere that helps researchers share analytic methods and computational models. Attendees will integrate their analytic methods or models into LearnSphere's Tigris in advance of the tutorial, and these methods will be made accessible to all tutorial attendees. We will learn about these different analytic methods during the tutorial and spend hands-on time applying them to a variety of educational datasets available in LearnSphere's DataShop. Finally, we will discuss the bottlenecks that remain, and brainstorm potential solutions, in openly sharing analytic methods through a central infrastructure like LearnSphere. Our goal is to create the building blocks to allow groups of researchers to integrate their data with other researchers to advance the learning sciences as harnessing and sharing big data has done for other fields.

Keywords

Learning metrics; data storage and sharing; data-informed learning theories; modeling; data-informed efforts; scalability.

1. INTRODUCTION

The use of data to improve student learning has become more effective as student learning activities and student progress through educational technologies are increasingly being tracked and stored. There is a large variety in the kinds, density, and volume of such data and to the analytic and adaptive learning methods that take advantage of it. Data can range from simple (e.g., clicks on menu items or structured symbolic expressions) to complex and harder-to-interpret (e.g., free-form essays, discussion board dialogues, or affect sensor information). Another dimension of variation is the

time scale in which observations of student behavior occur: click actions are observed within seconds in fluency-oriented math games or in vocabulary practice, problem-solving steps are observed every 20 seconds or so in modeling tool interfaces (e.g., spreadsheets, graphers, computer algebra) in intelligent tutoring systems for math and science, answers to comprehension-monitoring questions are given and learning resource choices are made every 15 minutes or so in massive open online courses (MOOCs), lesson completion is observed across days in learning management systems, chapter/unit test results are collected after weeks, end-of-course completion and exam scores are collected after many months, degree completion occurs across years, and long-term human goals like landing a job and achieving a good income occur across lifetimes. Different paradigms of data-driven education research differ both in the types of data they tend to use and in the time scale in which that data is collected. In fact, relative isolation within disciplinary silos is fostered and fed by differences in the types and time scale of data used (cf., Koedinger et al., 2012).

Thus, there is a broad need for an overarching data infrastructure to not only support sharing and use within the student data (e.g., clickstream, MOOC, discourse, affect) but to also support investigations that bridge across them. This will enable the research community to understand how and when long-term learning outcomes emerge as a causal consequence of real-time student interactions within the complex set of instructional options available (cf., Koedinger et al., 2010). Such an infrastructure will support novel, transformative, and multidisciplinary approaches to the use of data to create actionable knowledge to improve learning environments for STEM and other areas in the medium term and will revolutionize learning in the longer term.

LearnSphere transforms scientific discovery and innovation in education through a scalable data infrastructure designed to enable educators, learning scientists, and researchers to easily collaborate over shared data using the latest tools and technologies. LearnSphere.org provides a hub that integrates across existing data silos implemented at different universities, including educational technology "click stream" data in CMU's DataShop (Stamper et al., 2011), massive online course data in Stanford's DataStage and

Kenneth Koedinger, John Stamper, Paulo Carvalho, Philip Pavlik and Luke Eglington "Sharing and Reusing Data and Analytic Methods with LearnSphere" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 773 - 774

analytics in MIT's MOOCdb (Veeramachaneni et al., 2014), and educational language and discourse data in CMU's new DiscourseDB (Jo et al., 2016). LearnSphere integrates these DIBBs in two key ways: 1) with a web-based portal that points to these and other learning analytic resources and 2) with a web-based workflow authoring and sharing tool called Tigris. A major goal is to make it easier for researchers, course developers, and instructors to engage in learning analytics and educational data mining without programming skills.

The main goal of this tutorial is to provide attendees with hands-on experience using Tigris for learning analytics. We hope that this year we will be able to attract attendees that have been exposed to LearnSphere from these past events, although we will have some tutorial activities included for new attendees as well. This tutorial builds off a successful LAK 2018 Tutorial, and tutorial at AIED/EDM 2017.

Activities will include presentations from tutorial organizers. Hands-on sessions will include demos and group work towards implementing analytics. Broadly, this tutorial offers those in the Learning Analytics community exposure to LearnSphere as a community-based infrastructure for educational data and analysis tools. In opening lectures, the organizers will discuss the way LearnSphere connects data silos across universities and its unique capabilities for sharing data, models, analysis workflows, and visualizations while maintaining confidentiality.

More specifically, we propose to focus on attracting, integrating, and discussing researcher contributions to Tigris, the web-based workflow authoring and sharing tool. tutorial submissions in the form of abstracts will involve a brief description of an analysis pipeline relevant to modeling educational data as well as accompanying code. Prior to the tutorial itself, the organizers will coordinate with attendees to integrate their code into Tigris. A significant portion of the tutorial will be dedicated to a hands-on exploration of custom workflows and workflow modules within Tigris. Attendees will present their analysis pipelines, and everyone attending the tutorial will be able to access those analysis pipelines within Tigris to a variety of freely available educational datasets

available from LearnSphere. The goal is to generate -- for each workflow component contribution in the tutorial -- a publishable tutorial paper that describes the outcomes of openly sharing the analysis with the research community.

Finally, tutorial attendees will discuss bottlenecks that remain toward our goal of a unified repository. We will also brainstorm possible solutions. Our goal is to create the building blocks to allow groups of researchers to integrate their data with other researchers we can advance the learning sciences as harnessing and sharing big data has done for other fields.

2. ACKNOWLEDGMENTS

This workshop is supported by National Science Foundation grant ACI-1443068 toward the creation of LearnSphere.org

3. REFERENCES

- [1] Jo, Y., Tomar, G., Ferschke, O., Rosé, C. P., & Gašević, D. (2016, April). Pipeline for expediting learning analytics and student support from data in social learning. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge* (pp. 542-543). ACM.
- [2] Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2010). The knowledge-learning-instruction (KLI) framework: Toward bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*.
- [3] Stamper, J., Koedinger, K.R., Baker, R., Skogsholm, A., Leber, B., Demi, S., Yu, S., Spencer, D. (2011) Managing the Educational Dataset Lifecycle with DataShop. In Kay, J., Bull, S. and Biswas, G. (eds). *Proceeding of the 15th International Conference on Artificial Intelligence in Education* (AIED2011).
- [4] Veeramachaneni, K., Halawa, S., Dernoncourt, F., O'Reilly, U. M., Taylor, C., & Do, C. (2014). Moocdb: Developing standards and systems to support MOOC data science. *arXiv preprint*. arXiv:1406.2015.

EDM & Games: Leveling Up Engaged Learning with Data-Rich Analytics

Jonathan Rowe
North Carolina State University
Raleigh, NC 27695
jprowe@ncsu.edu

Bradford Mott
North Carolina State University
Raleigh, NC 27695
bwmott@ncsu.edu

Luc Paquette
University of Illinois at
Urbana-Champaign
Champaign, IL 61820
lpaq@illinois.edu

Seung Lee
North Carolina State University
Raleigh, NC 27695
sylee@ncsu.edu

ABSTRACT

There is growing evidence that digital games are an effective medium for learning. Games also present distinctive challenges and opportunities for educational data mining (EDM). Many games feature complex, open-ended tasks and unexpected learner behaviors. In addition, games provide vast amounts of fine-grained data about learners' behaviors, offering a new window into student learning processes. Game-based learning raises questions that require new computational techniques and methodologies to enhance our understanding of the moment-to-moment signatures of engaged learning. To explore these questions, we have convened a workshop on EDM & Games: Leveling Up Engaged Learning with Data-Rich Analytics. The purpose of the workshop is to build a community with shared interest in this emerging area, foster future collaborations, and identify new opportunities for EDM that are enabled by data from student interactions with game-based learning environments. The half-day workshop features a program of oral paper presentations, group discussions, and an invited speaker, each exploring questions about how EDM and games can be leveraged together to enhance our understanding of engaged learning in a range of educational contexts.

Keywords

Game-based learning, serious games, game-based assessment, multimodal analytics, engagement.

1. INTRODUCTION

Recent advances in educational data mining (EDM) are creating new opportunities for game-based learning. Games have long been recognized for their capacity to engage players in immersive virtual worlds and complex problem scenarios. Now, there is growing evidence that games are also an effective medium for enhancing learning [1, 2]. Games for learning have

been developed across a broad range of educational domains, including science education [3, 4], mathematics education [5], language learning [6], civic education [7], and patient education [8].

As learners interact with educational games, rich streams of in-game behavior data are generated that can be modeled and analyzed using EDM techniques. These data streams contain timestamped records of key game events, such as interactions with non-player characters, movements between locations, and manipulations of virtual objects. Increasingly, they also include multimodal sensor data, such as eye tracking, facial expression, posture, and biometric data [9, 10]. The increasing availability of trace-level data from games has created new opportunities for investigating methods to analyze and model learner behavior in these environments. The resulting models hold promise for enriching our understanding of learning, problem solving, and engagement [11, 12], as well as enabling data-driven pedagogical functionalities to create personalized game-based learning experiences [13].

In light of these advances, a variety of questions have begun to emerge at the intersection of EDM and games:

- Which EDM techniques are best suited for addressing the distinctive challenges posed by game-based learning, such as open-ended tasks, unexpected learner behaviors, and balancing learning and engagement effectively?
- How can we leverage EDM techniques to inform evidence-based design principles for game-based learning?
- What are the unique contributions that EDM can provide toward enhancing our understanding of the moment-to-moment signatures of engaged learning.

2. WORKSHOP GOALS

To explore these questions, the workshop brings together researchers interested in the intersection of EDM and games to better understand student learning and engagement using data-rich analytics. It showcases recent research on EDM in games for learning, training, and education. A key focus is leveraging novel data mining techniques and methodologies to enhance our understanding of student learning and engagement, as well as to enable enriched learning experiences that are both effective and engaging.

Jonathan Rowe, Bradford Mott, Luc Paquette and Seung Lee "EDM & Games: Leveling Up Engaged Learning with Data-Rich Analytics" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 775 - 776

The workshop covers a broad range of topics at the intersection of EDM and games across different educational settings (e.g., schools, training, online learning), student populations (e.g., primary, secondary, post-secondary), and educational domains (e.g., science, mathematics, military training). It has been designed to build a community of researchers that share common interests and to create opportunities for future collaborations. It has also seen the assembly of a multidisciplinary program committee consisting of more than 20 scholars spanning academia and industry, several of whom have also contributed to the workshop as authors.

3. WORKSHOP PROGRAM

The half-day workshop is structured around a series of oral paper presentations, group discussions, and an invited talk. Accepted papers cover a range of topics at the intersection of EDM and games, including the following:

- An investigation of problem-solving strategies of both expert humans and AI agents playing Atari 2600 games, including visualizations of game data to compare agent policies and human expert strategies.
- An examination of off-the-shelf data-mining techniques to predict student quitting and assessment performance with two online middle school science learning games.
- An investigation of how much data is needed in gameplay sessions to predict student performance, and thus inform how teachers should allocate help and attention toward students. Teachers may not always be able to attend to information dashboards in the classroom, so understanding how to alert teachers of student needs is critical.
- An approach to detecting student affect using multimodal sensors in game-based learning environments. The authors examine two sensor modalities, posture and electrodermal activity, to predict the affective states of students and evaluate different types of multimodal data fusion techniques to obtain the best classification accuracy.
- A data-driven approach to predicting students' challenge outcomes using a game-based learning environment for genetics. The model predicts two categories of challenge outcome: quit or complete. The paper explores and compares an extensive set of machine learning models for predicting student performance.

4. CONCLUSION

Game-based learning environments have significant potential as a medium for engaging learners in complex problem-solving scenarios. The rich data streams produced by student interactions with games are creating new opportunities for applying EDM to enrich our understanding of learning, problem solving, and engagement. This workshop represents an important step toward realizing this vision by cultivating a scholarly community that seeks to understand how we might foster engaged learning through the integration of EDM and game-based learning technologies and methods.

5. REFERENCES

- [1] Clark, D. B., Tanner-Smith, E. E., and Killingsworth, S. S. 2016. Digital games, design, and learning: A systematic review and meta-analysis. *Review of Educational Research*. 86, 1, 79–122.
- [2] Wouters, P., van Nimwegen, C., van Oostendorp, H., and van der Spek, E. D. 2013. A meta-analysis of the cognitive and motivational effects of serious games. *Journal of Educational Psychology*. 105, 2, 249–265.
- [3] Rowe, J., Shores, L., Mott, B., and Lester, J. 2011. Integrating learning, problem solving, and engagement in narrative-centered learning environments. *International Journal of Artificial Intelligence in Education*, 21, 1-2, 115–133.
- [4] Kim, Y. J., Almond, R. G., and Shute, V. J. 2016. Applying Evidence-Centered Design for the development of game-based assessments in Physics Playground. *International Journal of Testing*, 16, 2, 142–163.
- [5] Kim, H. and Ke, F. 2017. Effects of game-based learning in an OpenSim-supported virtual environment on mathematical performance. *Interactive Learning Environments*, 25, 4, 543–557.
- [6] Johnson, W. L. 2010. Serious use of a serious game for language learning. *International Journal of Artificial Intelligence in Education*, 20, 2, 175–195.
- [7] Easterday, M. W., Aleven, V., Scheines, R., and Carver, S. M. 2016. Using tutors to improve educational games: A cognitive game for policy argument. *Journal of the Learning Sciences*, 26, 2, 226–276.
- [8] Lu, A. S. and Kharrazi, H. 2018. A state-of-the-art systematic content analysis of games for health. *Games for Health Journal*, 7(1), 1–15.
- [9] Bosch, N., D'Mello, S. K., Baker, R. S., Ocumpaugh, J., Shute, V. J., Ventura, M., Wang, L., and Zhao, W. 2016. Detecting student emotions in computer-enabled classrooms. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence* (New York, NY, July 9–15, 2016). IJCAI '16. AAAI Press, Menlo Park, CA, 4125–4129.
- [10] Taub, M., Mudrick, N., Azevedo, R., Millar, G., Rowe, J., and Lester, J. 2017. Using multi-channel data with multi-level modeling to assess in-game performance during gameplay with Crystal Island. *Computers in Human Behavior*, 76, 641–655.
- [11] Bauer, A., Flatten, J., and Popovic, Z. 2017. Analysis of problem-solving behavior in open-ended scientific-discovery game challenges. In *Proceedings of the 10th International Conference on Educational Data Mining* (Wuhan, China, June 25–28, 2017). EDM '17. 32–39.
- [12] Akram, B., Min, W., Wiebe, E., Mott, B., Boyer, K. E., and Lester, J. 2018. Improving stealth assessment in game-based learning with LSTM-based analytics. In *Proceedings of the Eleventh International Conference on Educational Data Mining* (Buffalo, NY, July 15–18, 2018), EDM '18. 208–218.
- [13] Karumbaiah, S., Baker, R.S., and Shute, V. 2018 Predicting Quitting in Students Playing a Learning Game. In *Proceedings of the 11th International Conference on Educational Data Mining* (Buffalo, NY, July 15–18, 2018). EDM '18. 21

Causal Discovery with Tetrad in LearnSphere's Tigris

Richard Scheines
Department of Philosophy
Carnegie Mellon University
scheines@cmu.edu

Peter Schaldenbrand
Human-Computer Interaction
Institute
Carnegie Mellon University
pschalde@cs.cmu.edu

Kenneth Koedinger
Human-Computer Interaction
Institute
Carnegie Mellon University
koedinger@cmu.edu

ABSTRACT

This tutorial explores causal discovery algorithms in Tetrad¹ implemented in LearnSphere's Tigris workflow tool². The Tetrad software suite contains algorithms that search for causal models from observational and experimental datasets [10], and has been productively applied to many educational datasets [8, 6, 3]. LearnSphere's Tigris is a free online workflow authoring tool and data mining infrastructure for custom analyses of new and existing data formats, including the educational data repository DataShop [11]. The tutorial teaches the fundamentals of causal discovery with hands on work in Tetrad, and teaches integrated causal data-mining with hands on work in Tigris. Attendees will gain experience sharing their results and methods in Tigris with others as well as connecting their analyses to thousands of datasets available in DataShop.

1. INTRODUCTION

As more educational technologies are created, more data is being logged and available to researchers. Mining this data for information concerning what pattern of student behaviors cause better learning outcomes is crucial if we are to intervene, either in the design of the online material, or in the student's behavior more directly. The gold standard for determining causal relationships is randomized controlled experiments, but these cost money and time, and are often unable to answer questions about the mechanisms by which an intervention causes learning. Tetrad contains algorithms that can mine both experimental and observational datasets for information about the causal relationships that might have produced the observed data. It has been used to guide follow-up experiments in educational research [5]. With an abundance of datasets available and causal questions at the center of all kinds of science, educational data-mining for causation is an increasingly important task.

¹<http://www.phil.cmu.edu/tetrad/>

²<https://pslcdatashop.web.cmu.edu/LearnSphere>

Richard Scheines, Peter Schaldenbrand and Kenneth Koedinger "Causal Discovery with Tetrad in LearnSphere's Tigris" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 777 - 779

Tetrad is a standalone Java program developed by the Philosophy department at Carnegie Mellon University and more recently in conjunction with the Department of Biomedical Informatics at the University of Pittsburgh. The aim of the program is to provide sophisticated methods of creating, searching for, estimating, and testing causal and statistical models of both experimental and observational data. No prior programming experience is needed to use Tetrad.

Much of the functionality of Tetrad has been included in an online workflow tool called Tigris. The LearnSphere project created Tigris to connect multiple data sources to analytical tools that are open source and available for collaboration. LearnSphere.org provides a hub that integrates existing data silos implemented at different universities, including educational technology "click stream" data in CMU's DataShop[11], massive online course data in Stanford's DataStage and analytics in MIT's MOOCdb [12], and educational language and discourse data in CMU's DiscourseDB [1]. Researchers can share their data mining code in the Tigris tool by adding it to the open source repository³ such as the Performance Factors Analysis[4] program or the Feature Extractor [13].

The advantages of the Tetrad implementation in Tigris are numerous. Since Tigris is a web based tool, workflows can be shared amongst anyone with a computer. There is no need to worry about having Java on your computer or downloading different versions of Tetrad. The owner of a workflow can share their Tetrad analyses on Tigris with whomever they wish. In addition to shareability, Tetrad on Tigris is connected to the educational data repository DataShop. This connection allows for easy access and permission control of data in a Tetrad analysis.

Tigris workflows are executed and saved remotely on secure servers. LearnSphere has a scalable infrastructure that allows for computation on large datasets, and so the limitations on computation are not influenced by the user's machine. The analyses are run remotely, so the code still runs regardless of the status of the user's machine. The analyses can be accessed from any computer and are safely saved.

In this tutorial, we will explore the causes of learning in online courses using Tetrad in Tigris. The organizers will explain how to interpret the output generated by algorithms in Tetrad and how to build a Tigris workflow that utilizes data in DataShop to determine the causes of learning with

³<https://github.com/LearnSphere/WorkflowComponents>

Tetrad. The attendees will gain access to a dataset that was collected from an online college course. The data contains variables such as pages viewed, activities performed, quiz scores, and final exam grades. Using this data, the attendees will generate causal models in a similar fashion to [2] and [3].

2. THE TUTORIAL

2.1 Schedule

Table 1: Tutorial Schedule: July 2nd, 2019

Time	Item
1:00pm	Introduction and Definition of Causal Discovery and Tetrad
2:15pm	Demonstration: Building a Tigris workflow that utilizes DataShop data and Tetrad
2:45pm	Break
3:00pm	Hands-on: Determine the causes of learning using DataShop data in Tigris.
4:00pm	Closing / High Level Discussion

2.2 Introduction and Definition of Causal Discovery and Tetrad

Richard Scheines is the Dean of the Mariana Brown Dietrich College of Humanities and Social Sciences at Carnegie Mellon University and a Professor in the Department of Philosophy. His research is on causal discovery, and in particular, the problem of learning about causation from statistical evidence. The theoretical and computational dimensions of Dr. Scheines's work are implemented in the Tetrad causal discovery tool. In this first segment of the tutorial, Dr. Scheines will introduce the theory behind statistical causality and teach attendees about graph theory. This will give attendees the background necessary to understand the underlying workings of the causal discovery algorithms in Tetrad.

2.3 Demonstration: Building a Tigris workflow that utilizes DataShop data and Tetrad

Peter Schaldenbrand is a developer on the LearnSphere project and researcher in the Human-Computer Interaction Institute at Carnegie Mellon University. He has authored many analysis components in the Tigris workflow tool and has integrated Tetrad's causal search algorithms into Tigris. Mr. Schaldenbrand will introduce attendees to the online educational data mining tool, Tigris, in a demonstrative way in this segment of the tutorial. Attendees will learn how to view another researcher's workflows and even build off of them while using their own data. The connection between DataShop data and the analytical tools in Tigris will be demonstrated. Mr. Schaldenbrand will introduce many of the educational data mining tools implemented in Tigris including the Tetrad suite. Lastly, Mr. Schaldenbrand will build a workflow which will setup the attendees to build their own workflows in the hands on section.

2.4 Hands-on: Determine the causes of learning using DataShop data in Tigris.

Tutorial attendees will gain access to two datasets in DataShop. Both datasets were generated from Massive Open Online Courses, but the courses were for different subjects

and took place at different universities. Each dataset contains variables that measure pre- and post-test assessment as well as resource usage such as videos watched, activities completed, and pages read.

Attendees will import the data directly from DataShop into a new Tigris workflow which they have created. They will learn to clean the data for usage with the Tetrad components. They will then use the Tetrad components to explore the causes of learning within these MOOC environments. This analysis will be similar to that performed in [2] and [3].

3. OBJECTIVES AND OUTCOMES

One of the objectives of this tutorial is to introduce the importance of causal analysis to the educational data mining community. Many papers focus on finding interesting correlations in data such as [7] and [9]. These correlations can be fascinating but only give small clues into how to make actionable interventions in curricula, intelligent tutoring systems, or classrooms to increase learning. This tutorial will demonstrate that important causal relationships can be mined from observational data and that these relationships can give insight into real interventions on student learning.

This tutorial will also display the importance of making analyses sharable and introduce easily reproducible workflow capabilities in Tigris. Software versioning, data permissions, and hidden calculations plague the reproducibility of research results. In Tigris, versioning is hidden from the user allowing anyone with a computer to use the tools. Data permissions are handled through the safe infrastructure of DataShop, and all calculations and parameters are visible in the Tigris workflow pipeline. This tutorial will demonstrate that doing research in Tigris is a great way to support reproducibility and shareability.

One outcome of this tutorial is that attendees will be familiar with using data from DataShop in a Tigris workflow. They will request access to a dataset which they will import into a Tigris workflow under the guidance of the organizers. They will then gain familiarity with the Tetrad algorithms and analyze the results with the organizers.

Once the attendees are familiar with creating causal models in a workflow, they can move onto working on their own research questions. Tutorial attendees will discuss what causal relationships they would like to investigate and try to expand on the use cases of Tetrad. They may present a workflow that utilizes Tetrad and their own datasets to the other attendees. This tutorial will promote the importance of causal reasoning in education with the intention of making productive interventions on student learning.

4. REFERENCES

- [1] Y. Jo, G. Tomar, O. Ferschke, C. P. Rosé, and D. Gašević. Pipeline for expediting learning analytics and student support from data in social learning. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge, LAK '16*, pages 542–543, New York, NY, USA, 2016. ACM.
- [2] K. R. Koedinger, J. Kim, J. Z. Jia, E. A. McLaughlin, and N. L. Bier. Learning is not a spectator sport: Doing is better than watching for learning from a mooc.

- In G. Kiczales, D. M. Russell, and B. P. Woolf, editors, *Proceedings of the Second (2015) ACM Conference on Learning @ Scale, L@S '15*, pages 111–120, New York, NY, USA, 2015. ACM.
- [3] K. R. Koedinger, R. Scheines, and P. Schaldenbrand. Is the doer effect robust across multiple data sets? In *Proceedings of the 11th International Conference on Educational Data Mining, Buffalo, NY, July 16-20, 2018*, 2018.
 - [4] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis –a new alternative to knowledge tracing. In *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*, pages 531–538, Amsterdam, The Netherlands, The Netherlands, 2009. IOS Press.
 - [5] M. A. Rau and R. Scheines. Searching for variables and models to investigate mediators of learning from multiple representations. In *EDM*, pages 110–117. www.educationaldatamining.org, 2012.
 - [6] M. A. Rau, R. Scheines, V. Aleven, and N. Rummel. Does representational understanding enhance fluency - or vice versa? searching for mediation models. In *EDM*, pages 161–168. International Educational Data Mining Society, 2013.
 - [7] M. San Pedro, R. Baker, A. Bowers, and N. T. Heffernan. Predicting college enrollment from student interaction with an intelligent tutoring system in middle school. In *Proceedings of the Sixth International Conference on Education Data Mining, EDM 13*, 2013.
 - [8] R. Scheines, G. Leinhardt, J. Smith, and K. Cho. Replacing lecture with web-based course materials. *Journal of Educational Computing Research*, 32(1):1–26, 2005.
 - [9] S. Slater, J. Ocumpaugh, R. S. Baker, P. Scupelli, P. S. Inventado, and N. T. Heffernan. Semantic features of math problems: Relationships to student learning and engagement. In *In Proceedings of the 9th International Conference on Educational Data Mining*, pages 223–230. International Educational Data Mining Society (IEDMS), 2016.
 - [10] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT press, 2nd edition, 2000.
 - [11] J. C. Stamper, K. R. Koedinger, R. S. J. de Baker, A. Skogsholm, B. Leber, S. Demi, S. Yu, and D. Spencer. Managing the educational dataset lifecycle with datashop. In *Proceeding of the 15th International Conference on Artificial Intelligence in Education*, volume 6738 of *Lecture Notes in Computer Science*, pages 557–559. Springer, 2011.
 - [12] K. Veeramachaneni, S. Halawa, F. DERNONCOURT, U. O’Reilly, C. Taylor, and C. B. Do. Moocdb: Developing standards and systems to support MOOC data science. *Computing Research Repository*, abs/1406.2015, 2015.
 - [13] K. Veeramachaneni, U. O’Reilly, and C. Taylor. Towards feature engineering at scale for data from massive open online courses. *Computing Research Repository*, abs/1407.5238, 2014.

Author Index

Abdi, Solmaz	228
Afzal, Shazia	432
Aggarwal, Varun	444
Aguerreberere, Cecilia	234, 462
Ai, Fangzhe	240
Ait Khayi, Nisrine	246
Aleven, Vincent	468
Alexandron, Giora	129, 480
Ali Yahya, Anwar	484
Alstrup, Stephen	306, 572
Ambrose, G. Alex	432
An, Dongwook	336
Anderson, Henry	488
Andres-Bray, Juan Miguel	492
Angert, Tyler	496
Arbanowski, Stefan	599
Arnav, Ayush	474
Arthurs, Noah	252
Askinadze, Alexander	500
Aubert, Olivier	390
Aulck, Lovenoor	9
Azevedo, Roger	8
Azizoltani, Hamoon	168, 595, 766
Baba, Yukino	426
Bachelet, Rémi	564
Bahargam, Sanaz	504
Bailey, James	354
Baker, Ryan S.	49, 208, 378, 488, 492, 508, 732
Banjade, Rajendra	258
Bar-Ilan, Ori	659
Baraniuk, Richard	667
Barmaki, Roghayeh	695
Barnes, Tiffany	7, 59, 119, 168, 218, 595, 679, 766
Bechger, Timo	372
Ben-David, Yossi	659
Bhatnagar, Sameer	556
Bidelman, Gavin	516
Bittencourt, Ig Ibert	438
Blumenstock, Joshua	9
Boodhwani, Afshan	488
Botelho, Anthony F.	508, 663
Bouchet, François	312, 564
Boulanger, David	512
Boumi, Shahab	264
Bourda, Yolaine	29
Bouthillier, Iris Bourgault	768
Bouwers, Eric	760
Boyer, Kristy	39
Bradford, Allison	675, 718
Breitfeller, Luke	330
Broekman, Francette	760
Broisin, Julien	270
Brooks, Christopher	49
Bryant, Tonya	318
Bulger, Monica	234
Burnett, Vesper	378
Buttery, Paula	360, 687
Béland, Sébastien	768

Caines, Andrew	360, 687
Cao, Meng	516
Capello, Sarah	420
Capoor, Brahm	294
Cardoso-Leite, Pedro	548, 615
Carvalho, Paulo	773
Chawla, Nitesh V.	432
Chen, Guanliang	19
Chen, Lujie	520
Chen, Yishuai	240
Chen, Zhongzhou	560
Chetlur, Malolan	432
Chi, Min	119, 168, 595, 766
Chng, Edwin	639
Choffin, Benoît	29
Chopra, Shivangi	524
Christie, S. Thomas	726
Circi, Ruhan	568
Cobo, Cristóbal	234
Cohen, Anat	603
Coleman, Chad	732
Conrad, Stefan	500
Cristea, Alexandra	438
d'Aquin, Mathieu	188
D'Mello, Sidney K.	79, 324, 420
Dai, Huan	691
Dale, Meghan	420
Dalton, Elizabeth	770
Dang, Steven	276
Davis, Andrea	528
Davis, Christopher	687
Davis, Glenn	532
de Barba, Paula	354
Dede, Chris	643
Desmarais, Michel	556, 611, 768
Dickler, Rachel	536
Ding, Xinyi	282
Doan, Thanh-Nam	288
Dong, Matthew	540
Dong, Yihuan	218
Donnelly, Patrick	420
Douglas, Kerrie	619
Dubrawski, Artur	520
Duckworth, Angela L.	79
Dufresne, Aude	623
Dziuban, Charles	742
E. Mello, Carlos	627
Eglington, Luke G.	5, 773
Elliott, Mark	360
Emerson, Andrew	39
Emond, Bruno	544
Ezra, Orit	603
Fancsali, Stephen	468
Fang, Weiqi	468
Farrar, Scott	342
Ferreira, Rafael	19
Flemming, Rory	548, 615
Fu, Guowei	240
Furr, Daniel	528, 552

Gagnon, Vincent	556
Gal, Kobi	408, 603
Gao, Jing	450, 671
Garcia, Julian	584
García, Sofía	234
Gardner, Josh	49
Gardner, Margo	79
Gasevic, Dragan	19, 228, 414, 651
Gehringer, Edward	474
Gerber, Sue	584
Gilad-Bachrach, Ran	659
Giles, C. Lee	348
Gitinabard, Niki	59, 679
Gjekmarkaj, Eva	520
Gobert, Janice	536
Godley, Amanda	420
Golab, Lukas	524
Gong, Jiaqi	607
Grimaldi, Phillip	667
Gross, Markus	198
Gross, Sebastian	7
Gruver, Nate	294
Guo, Yuchun	240
Guo, Zhang	695
Guthrie, Matthew	560
Hammer, Barbara	7
Han, Yong	300
Hansen, Casper	306
Hansen, Christian	306
Harmon, Jon	738
Harpstead, Erik	456
Harrah, Fatima	312, 564
Hatala, Marek	651
Heckman, Sarah	59
Heffernan, Neil T.	208, 508, 663
HersHKovitz, Arnon	603
HersHKovitz, Sara	129
Hicks, Juanita	568
Hjuler, Niklas	572
Hoernle, Nicholas	408
Hofman, Abe	372
Holstein, Kenneth	468
Howlin, Colm	742
Hu, Qian	69
Huang, Karina	318
Huang, Yanzun	468
Hutt, Stephen	79, 324
Hylak, Benjamin	462
Hérouard, Clément	270
Iandoli, Matthew	89
Ikedu, Mizuho	576
Isotani, Seiji	438
Jacobs, Daniel	378
Jacobsen, Anik	580
Jarratt, Daniel	726
Jayaraman, J.D	584
Jensen, Emily	324
Jeon, Byungsoo	330
Ji, Suozhao	300

Jiang, Han	89
Jiang, Nan	588
Jiang, Tianrui	635
Jing, Xia	450
Jo, Jaechoon	591
Ju, Song	595
Kang, Jina	336
Kaplan, Gabriela	234
Karayev, Sergey	252
Karumbaiah, Shamyia	99
Karypis, George	6, 366, 384, 396
Kashima, Hisashi	426
Kelly, Sean	420
Kennedy, Gregor	354
Keylor, Eric	378
Khan, Abeer	524
Khoshnevisan, Farzaneh	119
Khosravi, Hassan	228
Khurana, Gauraang	474
Kim, Gyeongmin	591
Kim, Yeo Jin	766
Klingler, Severin	198
Koedinger, Kenneth R.	276, 456, 773, 777
Kolb, John	342
Konomi, Shin'Ichi	655
Konomi, Shinichi	178
Krauss, Christopher	599
Kumar, Vivekanandan	512
Käser, Tanja	109
Labrie, Audrey	556
Lan, Andrew	158, 208, 667
Lang, David	19
Lappas, Theodoros	504
Larson, Eric	282
Lee, Seung	775
Leite, Walter	402
Lester, James	39
Levin, Jason	330
Levy, Ben	603
Li, Haiying	536
Li, Mengyi	568
Li, Qi	450
Liang, Chen	348
Lim, Heuiseok	591
Linn, Marcia	675, 718
Lioma, Christina	306
Liu, Min	336
Liu, Shichao	89
Loignon, Guillaume	768
Lopez, Adam	414
Lorenzen, Stephan	572
Loye, Nathalie	768
Lu, Dongyang	468
Lu, Xiwen	663
Lucas, Christopher G.	414
Luengo, Vanda	312, 564
Lukin, Galit	659
Lynch, Collin F.	59, 679
Lytle, Nicholas	218
Ma, Boxuan	699

Ma, Fenglong	671
MacLellan, Christopher	456
Malekian, Donia	354
Malik, Ali	294
Mandalapu, Varun	607
Mao, Ye	119
Marcoulides, Katerina	402
Maris, Gunter	372
Marwan, Samiha	218
Mbouzao, Boniface	611
McCurdy, Kate	414
McLaren, Bruce	139, 702
Merceron, Agathe	599
Min, Wookhee	39
Mirsafian, Melicaalsadat	524
Modiri, Arghavan	647
Mohammed, Fekri	484
Monllaó Olivé, David	770
Moore, Russell	360, 687
Morsomme, Raphael	748
Morsy, Sara	6, 366
Mott, Bradford	39, 775
Mouri, Kousuke	178
Mozer, Michael	4
Mussack, Dominic	548, 615
Nambi, Dev	9
Nanda, Gaurav	619
Nawaz, Sadia	354
Nazaretsky, Tanya	129
Nguyen, Huy	139, 702
Nikolakopoulos, Athanasios N.	396
Ning, Xia	158
Nkambou, Roger	623
Nyamen Tato, Ange Adrienne	623
Ocuppaugh, Jaclyn	99, 492
Ogata, Hiroaki	178
Oliveira, Mariana	627
Oliveira, Wilk	438
Olson, Lukas	726
Osman, Addin	484
Ou, Lu	372
Owen, V. Elizabeth	378
P. Ros ÌAe, Carolyn	330
Paaßen, Benjamin	7
Paepcke, Andreas	294
Pandey, Shalini	384
Paquette, Luc	775
Pardos, Zachary A.	342, 540, 588, 675, 718
Pavlik Jr., Philip I.	5, 516, 773
Piech, Chris	252, 294
Pigeau, Antoine	390
Pinkwart, Niels	7
Polyzou, Agoritsa	396
Popineau, Fabrice	29
Pramudianto, Ferry	474
Praseeda,	631
Price, Thomas	7, 119, 218
Pritchard, David	480
Prié, Yannick	390
Pu, Yanjun	635

Pursel, Bart	348
Raborn, Anthony	402
Rachatasumrit, Napol	456
Radhakrishnan, Balaji	474
Radu, Iulian	639
Rafferty, Anna	647, 766
Ram, Prasad	631
Ramirez, Lucia	639
Rangwala, Huzefa	69, 158
Reddick, Rachel	754
Reilly, Joseph	149, 643
Ren, Zhiyun	158
Rho, Yun Jin	528
Rice, Andrew	360, 687
Ritter, Steve	3
Rodríguez, Fernando	39
Ronen, Meitar	659
Rowe, Jonathan	775
Roy, Marie-Helene	378
Ruipérez-Valiente, José A.	480
Rus, Vasile	246, 258
S Baker, Ryan	99
Sadiq, Shazia	228
Sahebi, Shaghayegh	288
Sanz Ausin, Markel	168, 766
Schaldenbrand, Peter	777
Scheines, Richard	777
Schinazi, Victor	198
Schmück, Emmanuel	548
Schneider, Bertrand	149, 318, 496, 639
Schrater, Paul	548, 615
Schwartz, Daniel L.	109
Segal, Avi	603
Shafran, Eyal	330
Shaikh, Hammad	647
Shang, Xuequn	691
Sher, Varshita	651, 706
Shi, Lei	438
Shillo, Roi	408, 659
Shimada, Atsushi	178, 655
Shrier, Ian	611
Simmering, Vanessa	372
Sinclair, Arabella	414
Sitton, Daniel	659
Smeets, Roger	760
Smith, Andy	39
Smith, Cody	39
Solenthaler, Barbara	198
Sonmez Unal, Deniz	710
Spanakis, Gerasimos	580
Srinivasa, Srinath	631
Stamper, John	139, 468, 702, 773
Stenhaug, Ben	252
Stephenson, Shonte	732
Stevens, Mitchell	294
Stone, Cathlyn	420
Studer, Christoph	208
Sunahase, Takeru	426
Syed, Munira	432
Tabach, Michal	603

Taijala, Taavi	726
Takhar, Rohit	444
Tang, Jie	450
Taniguchi, Rin-Ichiro	178
Taniguchi, Yuta	178, 655
Taub, Michelle	8
Terzi, Evimaria	504
Thai, K. P.	378
Toda, Armando Maciel	438
Tzayada, Odelya	603
Unnam, Abhishek	444
Vainas, Oded	659
Valdés, Julio J.	544
van der Maas, Han	372
Van Dessel, Steven	89
Varatharaj, Ashvini	663
Vassileva, Julita	1
Vazquez Alferez, Sofia	748
Vela, Adan	264
Velagapudi, Nishant	9
Venant, Rémi	188
Vie, Jill-Jënn	29
Wampfler, Rafael	198
Wang, Cindy	532
Wang, Guangyan	240
Wang, Jingyu	468
Wang, Tianqi	450, 671
Wang, Yeyu	139
Wang, Zhenzhu	240
Wang, Zichao	667, 714
Warnakulasooriya, Rasil	738
Waters, Andrew	667
Weitekamp, Daniel	456
West, Jevin	9
Whitehill, Jacob	89, 234, 462
Wiebe, Eric	39
Wiley, Korah	675, 718
Williams, Joseph Jay	647, 766
Woodruff, Matthew	722
Wu, Wenjun	300, 635
Xiao, Yunkai	474
Xiao, Zhongcan	474
Xu, Yiqiao	679
Yan, Lili	336
Yang, Tsung-Yen	208
Yang, Yeongwook	591
Yang, Yuming	49
Yao, William	639
Ye, Jianbo	348
Yeung, Chun-Kit	683
Yu, Run	540
Yuan, Christina	532
Yun, Yue	691
Zaidi, Ahmed	360, 687
Zhang, Chuankai	468
Zhang, Hui	300
Zhang, Lijun	300

Zhang, Yupei	691
Zhao, Han	348
Zhao, Yongxiang	240
Zhi, Rui	119, 218
Zhou, Guojing	595
Zingle, Gabriel	474

Keyword Index

ABET accreditation	484
academic outcomes	264
Active learning	208
Active Learning	348
activity patterns	655
Adaptive Educational systems	228
Adaptive Learner Modeling	5
ADAPTIVE LEARNING	659
Adaptive Learning	129
adaptive learning	462, 754
adaptive spacing	29
Adaptive training	516
Additive Factors Model	588
Adversarial Deep Q-Network	595
affect detection	760
Affect Detection	324
Affect State	607
Affective Computing	198
agent	414
algorithmic bias	488
alignment	414
Analysis Reproducibility	777
analytics	627
Anomaly Detection	480
Answer Assessment	258
ant colony optimization algorithm	402
Apprentice Learner Architecture	456
Assessment	444
Association Analysis	699
association rule	484
At-risk students	584
ATC framework	635
attainment	722
attention networks	366
atypical behaviors	742
Audio	663
auto-grader	300
automated assessment	536
Automated Essay Evaluation	675, 718
Automated Essay Scoring	512
automated face analysis	89
Automated grading	444
Automated Student Assessment Prize	512
automatic evaluation	714
Automatic feedback	420
Automatic grading	667
Automatic Grading	663
Automatic Speech Recognition	420
Average Model	631
Bayesian model	300
behavior detection	378
bias selection	627
bidirectional encoder representation of the transformer	667
Binary weight learning	588
Biometric Sensors	198
Biometrics	496
Blended courses	59
Block-Based Programming	39

Bloom's Taxonomy	3
bot framework	722
C programming	655
Cardinal peer grading estimation	450
Causal Discovery	777
chatbot	414
Cheating	480
Classification	198
classification techniques	474
Classroom analytics	420
Classroom Discourse	420
Classroom Instruction	675, 718
Clickstream Data	330
Closing the loop between the research and practice	544
cluster analysis	552
Clustering	246, 324
clustering	414, 742
Clustering Educational Items	129
co-operative education	524
co-taken courses	158
Coding scheme	312, 564
Cognitive diagnosis	691
Cognitive model	635
Coh-Metrix	149
collaboration	149
collaborative filtering	603
Collaborative learning	695
collaborative learning analysis	695
Collaborative problem-solving	336
college	9
college applications	79
College success	79
Common App	79
community detection	679
Computational Theories of Learning	456
Computer Supported Collaborative Learning	496
computer-supported collaborative work	149
concept complexity	188
concept discovery	615
Concept Prerequisite Learning	348
Conceptual Relation Mining	675, 718
confirmatory factor analysis	402
content improvement	738
content pedagogy	738
content sequencing	603
Contextual Multi-armed Bandits	647
continuous-time model	372
Convolutional Neural Networks	623
Course Planning and Selection	294
Course Recommendation	6
course recommendation	396, 748
course search	540
course2vec	540
Creativity Detection	408
crowdsourcing	462
Crowdsourcing	702
Crucial Pedagogical Decision	595
cumulative knowledge	158
curriculum	770
dAFM	588
Data Hoarding	1

data mining	738
Data Mining	438, 699
data storage and sharing	773
Data Storage and Sharing	777
Data Visualization	496
data-driven	79, 218
data-driven approach	536
data-informed efforts	773
Data-Informed Learning Theories	777
data-informed learning theories	773
DataSet	258
Decimal Number	139
decision support	726
Decision Support	619
deep learning	119, 306, 508, 615, 643, 671, 683, 687
Deep learning	69
Deep Learning	240, 330, 512
Deep Reinforcement Learning	168
demographic analysis	492
Dialogic Instruction	420
dialogue	414
Difficulty level	516
Digital Learning Game	139
Digitally-Based Assessments	568
Dimensionality reduction	544
discovering curricular gaps	580
discussion forum	432
Discussion forum	564
disengaged behavior	508
Disengagement	710
drop out	9
Drop out prediction	500
Dropout	49
dropout prevention	726
dynamic difficulty adjustment	548
Dynamic Experimentation	647
E-LEARNING	659
e-textbooks	655
early prediction	119
early warning	726
EDAI	722
Edit distance	270
Education	444
Education settings	768
education testing	615
Educational Data Mining	607, 766, 770
educational data mining	484
Educational data mining	69, 270
Educational Dialogue Analysis	19
educational games	378
Educational measurements	768
Educational Recommender Systems	599
educational session modelling	306
Educational Systems	572
EFL	234
Elo-based models	228
Email Writing	444
Embedding	342
emotions	432
Empirical Models	4
engagement.	775
English as a foreign language	234

English as a second language	234
Enrollment Modeling	294
Ensemble learning	312
Ensembling	732
entropy	627
error resolution process	655
ESL	234
Essay Semantic Analysis	675, 718
Ethics	1
Evidence	631
Evidence Based Competency Model	631
Exam Scores	252
example-based feedback	218
Exercise Recommendation	240
expert evaluation	218
Expert Knowledge	623
Factor Analysis	607
fairness	488
Fairness Accountability & Transparency	1
Fairness in Machine Learning	79
feature engineering	508
feature extraction	390
Feature Extraction	663
Feature Selection	607
features engineering	188
Feedback	528
Feedback Timing	528
forum participation	679
fuzzy clustering	742
game-based assessment	775
Game-based learning	548, 775
Gamification	438, 591
Gaming	5
Gaussian Mixture Models	294
Gaussian Models	5
Gaussian Processes	168
gender	492
gender gap in STEM	524
Generalizability	79, 324
genetic algorithm	402
grade prediction	366
Gradient Tree Boosting	19
graduation	488
grammatical errors	687
Graph convolutional networks	69
graph-based methods	532
Heterogenous Treatment Effects	647
Hidden Markov Model	264
Hidden Markov Models	294
hierarchical structure modeling	671
High School Graduation	732
higher education	540, 627
Hint Generation	7
human in the loop	722
human-machine hybrid algorithm	300
Hybrid Learning Environments	699
Hybrid method	312
Hybrid Neural Networks	623
immersive virtual environment	643

Individual Differences	528
Individualization	631
inferred keywords	540
Instrumentation	639
intelligent tutoring system	468
intelligent tutoring systems	246
Intelligent Tutoring Systems	3, 7, 19, 129, 258, 276, 766
interaction features	99
interactive learning environment	552
interactive learning environments	548
IRT	252
item response models	738
item response theory	683, 738
item similarity	615
Jaccard coefficient	336
job matching	580
k-medoids	552
KC Model	139
Keystroke log	354
KmL cluster analysis	336
knowledge components	29
knowledge graph	188
knowledge representation	360
Knowledge Tracing	240, 282, 384
knowledge tracing	671, 683, 754
Knowledge-based Filtering	599
knowledge-based models	366
L-MMSE estimation	208
language learning	687
Language learning	414
Latent Dirichlet Allocation	294, 580
latent topics	540
LDA	330
Learner Model	623
Learner model	228
Learner Models	651
learning activity analytics	178
Learning Analytics	480, 639, 651, 706, 732
Learning analytics	584
learning behavior	109
Learning Curve Analysis	456
Learning Feature	576
Learning metrics	773
Learning Process	631
learning process	336
Learning programming	270
Learning Success	651
learning transfer	738
Levenshtein algorithm	270
log based metrics	760
Log Data Analysis	560
Logit-Normal	252
Long Short Term Memory	623
long short-term memory network	643
Lorenz Curve	576
machine learning	9, 79, 119, 488, 722, 726
Machine Learning	348, 619, 663, 732
Machine learning models	584
Makerspaces	639

Mandarin tone	516
Markov Chain	560
Markov chains	396
Markov Decision Processes	766
Mask R-CNN	695
Massive Open Online Courses	384
Massive Open Online Courses (MOOCs)	450
math identity	99
math problem solving	520
math success	99
matrix factorization	158
Measurement	276
memory	29
Mentor's Notes	330
meta-learning	667
Metacognition	8
Minority student retention	584
Misconception	342
mixed-effects model	372
Mixture Models	294
Mobile Eye-tracking	318
Mobile Learning	651, 706
model comparison	390
Model evaluation	49
modeling	773
Monitoring Tools	639
MOOC	390, 432, 492, 564, 679
MOOC Discussion Forum	619
MOOCs	49, 480, 611
Moodle	770
MORF	492
Motion Capture/Tracking	639
Motion Sensing	318
Motivation	276
Mouse Cursor Tracking	576
multi modal learning analytics	520
multi-modal learning analytics	149
multicomponent latent trait model	360
multimodal analytics	775
Multimodal Learning Analytics	318
Multinomial Logistic Regression	342
mutualism model	372
NAEP	568
National Student Clearinghouse	79
natural language processing	532, 667, 679
Natural Language Processing	420, 496, 512, 702
neural network	603
Neural Network	512
Neural networks	49, 588
Neural Networks	4
neural question generation	714
next-term grade prediction	158
Non-routine Mathematics	702
Normal Distribution	252
novice programming	119
object recognition	695
online classification	109
online discussion forums	532
Online Education	591
online feedback	234
Online Learning	5, 324, 528

Online Learning Modules	560
online tutors	99
open curriculum	748
open educational resources (OER)	462
open ended systems	408
open exploration environments	109
open systems	770
Open vocabulary	420
Open-ended Learning Environment	639
optimal scheduling	29
optimization	178
outliers	742
Partitioning	504
partitioning around medoids	552
Pedagogical Policy	168
pedagogy	770
Peer assessment	426
peer assessment	474
peer grading	300
Peer grading	426
performance	432
Performance factors analysis	516
Performance Prediction	69
perseverance	520
Personalised gamification	438
Personalized education	691
personalized learning	378
Personalized Learning	1, 384
PERSONALIZED LEARNING PATH	659
Persuasive Technologies	1
PFA	5
Physiological Sensing	318
Polytomous	252
Post-secondary Education	294
pre-requisite discovery.	748
prediction	468
Predictive Analytics	512
predictive analytics	722
predictive modeling	378
Predictive modeling	49
Predictive Student Models	39
priming	738
Probabilistic model	426, 450
Probabilistic Soft Logic	258
Process Data	568
productive persistence	378
program educational objectives	484
programming exercise	655
psychometric scales	402
Psychometrics	768
q-matrix	29
Q-matrix refinement	588
Qualitative Feedback	675, 718
Quality Improvement	576
Question categorization	312
question generation	714
race	492
random forest	79
random forests	468
random walks	396

Randomized Control Trials	647
rank model	288
re-enrollment	9
Recent Temporal Pattern Mining	119
reciprocal recommender systems	580
recommender systems	540
Recommender Systems	699
recurrent neural network	643
Recurrent Neural Network	282
recurrent neural networks	109
regression	548
Regression models	710
regular expressions	536
Reinforcement Learning	7, 595, 766
reliability	402
repeated fuzzy clustering	742
Replication	49
Representation Learning	6
Response time	710
Rubrics	512
scalability	773
school demographics	99
science learning	336
scientific explanations	536
second language	414
secondary education	234
Self-Regulated Learning	8
Self-regulation	276, 564
Semantic analysis	270
semantic complexity	188
Semi-structured Text analysis	444
Semi-Supervised Clustering	504
Sensor-free	324
Sequence analysis	59
Sequence Mining	8
Sequence Modeling	384
sequential recommendation	396
serious game	336
Serious Game	623
serious games	548, 775
SHAP feature importance	354
Short answer grading	556
short answer questions	667
short form selection	402
Short-Term Memory	591
Siamese Neural Network	572
Similarity Measurement	129
simulated annealing	402
Simulated Students	4
Skills dimension determination	635
skills embeddings	360
skip-gram	540
Skip-Gram	342
slicing analysis	79
social network analysis	679
Socio-Moral Reasoning skill	623
Sparse factor learning	691
Stacking	312
Stealth assessment	643
STEM	607
Student Ability	252
student ability estimation	754

student affect	508
student affect detection	208
Student assignment to courses	178
student behaviors	742
student behaviour	306
Student clustering	572
Student dropout	584
Student dropout prediction	584
student enrollment mode	264
Student grouping	691
Student Model	240
Student model	635
Student modeling	691, 710
student modeling	29, 468, 754
Student Modeling	6, 702
student modelling	414, 687
student outcomes	484, 722
student performance prediction	288
Student Performance Prediction	39
student representations	687
Student response systems	556
Student Satisfaction	19
Student State	330
students modelling.	246
students' learning	89
Student's performance	564
Student's question	312, 564
Study Sessions	59
Stylus	198
success prediction	390
suggestion mining	474
Supervised Learning	444
Supervised learning	49
Support Functions	568
SVM Kernel	500
tabu search	402
task representations	687
Team Formation	504
Temporal analysis	354
tensor factorization	288
Testing	252
text analytics	474
Text Classification	619
text mining	474
thermal comfort	89
Time Series	500
Time-Dependent Recommendations	599
Time-series Modeling	330
topic modeling	748
Trace Analysis	706
transcripts	9
treatment effect estimation	462
tutoring	414
Two-Sigma	3
Unsupervised Machine Learning	318
User Characteristics	438
User Modeling	599
valence	432
validity	402
Vector space model	556

Video interaction encoding	611
Video interaction patterns	611
Video Lecture	591
Visual analytics	544
Visualization	282
visualization	552
Visualizations	408
wheel-spinning	378, 468
Word Embeddings	675, 718
work-integrated learning	524
worked examples	218
Writing process	354
Writing style analysis	572
XGBoost	354
young students	760
ZONE OF PROXIMAL DEVELOPMENT	659